Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

ADVANCES IN DEEP LEARNING WITH APPLICATIONS TO COMPUTER VISION AND ASTRONOMY

by
Zhihang Hu

Deep Learning has spanned a variety of applications in computer vision as well as computational astronomy. These two aspects obtained similar data structure, therefore, their solutions can be transferable between each other. This dissertation look into two video-related tasks in computer vision and propose a novel problem in computational astronomy.

Specifically, acquiring an in-depth understanding of videos has been a cornerstone problem in computer vision. This problem has been studied by various researchers from different perspectives, among which video prediction has attracted much attention. Video prediction aims to generate the pixels of future frames given a sequence of context frames. In practice, unlabeled video sequences can be gathered autonomously from a sensor or recording device. A machine capable of predicting future events using these video sequences in an unsupervised manner will gain extensive and deep knowledge about its physical environment and surroundings. However, despite its appealing prospects, accurate video prediction remains an open problem. The major challenge is the inherent uncertainty in the dynamics of the world. A typical example is that the future trajectory of a ball hitting the ground is inherently random.

This dissertation proposes new generative adversarial networks (GANs) for stochastic video prediction. The proposed framework, dubbed Video-Prediction-GAN(VPGAN), employs an adversarial inference model and a cycle-consistency loss function to empower the framework to obtain more accurate predictions. In addition, a conformal mapping network structure is incorporated into VPGAN to enable action

control for generating desirable future frames. In this way, VPGAN can produce fake videos of an object moving along a specific direction. Experimental results based on different datasets demonstrate the good performance of VPGAN and its superiority over existing methods. Other contributions of this dissertation include the development of a Three-Dimensional atrous convolutional long short-term memory network for background subtraction used in video processing and an extension of VPGAN for synthesizing vector magnetograms of active regions in different solar cycles. The dissertation concludes by pointing out some directions of future research in applying deep learning to computer vision and astronomy.

ADVANCES IN DEEP LEARNING WITH APPLICATIONS TO COMPUTER VISION AND ASTRONOMY

by
Zhihang Hu

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Science

Department of Computer Science

August 2021

APPROVAL PAGE

ADVANCES IN DEEP LEARNING WITH APPLICATIONS TO COMPUTER VISION AND ASTRONOMY

Zhihang Hu

Dr. Jason T.L. Wang, Dissertation Advisor — Date
Professor of Computer Science, New Jersey Institute of Technology

---

Dr. Ioannis Koutis, Committee Member — Date
Associate Professor of Computer Science, New Jersey Institute of Technology

---

Dr. Jing Li, Committee Member — Date
Assistant Professor of Computer Science, New Jersey Institute of Technology

---

Dr. Katherine Herbert, Committee Member — Date
Associate Professor of Computer Science, Montclair State University

---

Dr. Haimin Wang, Committee Member — Date
Distinguished Professor of Physics, New Jersey Institute of Technology

This work is completely dedicated to my respectful parents Tong Man and GuiSheng Hu without whose constant support this dissertation paper was not possible. They always inspire me. At the same time, my thanks also go to my caring siblings whose advice really worked for this dissertation paper.

ACKNOWLEDGMENT

I would first like to thank my advisor Dr. Jason Wang for his help throughout my whole PhD career. He taught me how to do research, helped me with my studies and most importantly, guided me with my future directions.

Then, I really appreciate the committee members for joining my dissertation defense and provide me with many precious advice. During these four years of studies, I served as a teaching assistant and was supported by the Computer Science department, therefore, I would also really like to thank the department for supporting my studies and fulfilled my academic pursuit.

Finally, very grateful to all my colleagues Hang Shi, ShaoBo Liu, Ruqi Pei, my friends YiDi Li, and my family, my mother, Tong Man, my father, GuiSheng Hu, and also my uncle's family.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

Deep Learning [37][49] have achieved tremendous attention and rapid growth in the field of Artificial Intelligence recent years. In fact, it actually has a long-term history, and this third wave of neural networks research blast initiates from two major reasons.

First, the availability of faster GPU and better hardware-software infrastructure allows learning models to go deeper and larger capacity [**?**] which serve as two crucial factors in neuroscience(imitating human-brain structure).

Also, the availability of larger size datasets such as CIFAR-10 [48], ImageNet10k [17] enable the models to do richer and more accurate representative learning which reduces human hand-crafted impact and increases the generalization of learned features.

These advances naturally fit the common traits of computer vision and astronomy problems which are usually high dimensional, hand-crafted difficult. Thus, deep learning are widely applied on visual problems such as image segmentation, video manipulation, or astronomical image super-resolution, etc and quickly push the accuracy to human-level.

1.1 Motivation

Although deep learning techniques played as the state-of-the-art role in many standardized problems, they're still far from satisfactory. In this dissertation, we propose to look into three Computer vision - Astronomical related open questions.

1.1.1 Background Subtraction

Background subtraction, or foreground detection, is a challenging problem in video processing. This problem is mainly concerned with a binary classification task, which

designates each pixel in a video sequence as belonging to either the background or foreground scene. Traditional approaches [76][13] for tackling this problem lack the power of capturing deep information in videos from a dynamic environment encountered in real-world applications, thus often achieving low accuracy and unsatisfactory performance. Such dissatisfaction forbids the future application of automic video surveillance processing and requires huge human-labor to examine those video data in many scenarios. This requires us to come up with a solution that tackles these difficulties.

1.1.2 Video Prediction

The ability of predicting future frames in video sequences, known as video prediction, is an appealing yet challenging task in computer vision. This task requires an in-depth representation of video sequences and a deep understanding of real-word causal rules. Existing approaches [62][14] for tackling the video prediction problem can be classified into two categories: deterministic and stochastic methods. Deterministic methods lack the ability of generating possible future frames and often yield blurry predictions. On the other hand, although current stochastic approaches can predict possible future frames, their models lack the ability of action control in the sense that they cannot generate the desired future frames conditioned on a specific action. Thus, in this research, we desire to come up with a model that could do both accurate predictions and action control.

1.1.3 Magnetograms Prediction

Many important magnetic field parameters are derived from vector magnetograms. These parameters have been used to analyze and forecast solar flare activity. Unfortunately, the most recent solar cycle 24 has been relatively weak with few large flares, though it is the only solar cycle in which consistent vector magnetograms

are available through the Helioseismic and Magnetic Imager (HMI) aboard the Solar Dynamics Observatory (SDO). In this paper we look into another major instrument, namely the Michelson Doppler Imager (MDI) aboard the Solar and Heliospheric Observatory (SOHO). The data archive of SOHO/MDI covers more active solar cycle 23 with many large flares. However, SOHO/MDI data only has line-of-sight (LOS) magnetograms, without vector magnetograms. This motivates us to take advantage of deep learning's strong representation capability to generate artificial vector magnetograms for astronomical research.

1.2 Contributions

We've done many works to tackle the above open questions, our main contributions are concluded in three aspects in the following dissertation.

1.2.1 Three-Dimensional Atrous ConvLSTM Network

We introduce a new 3D atrous convolutional neural network (CNN), used as a deep visual feature extractor, and stack convolutional long short-term memory (ConvLSTM) networks on top of the feature extractor to capture long-term dependencies in video data. This novel architecture is named a 3D atrous convolutional long short-term memory network. The new network can capture not only deep spatial information but also long-term temporal information in the video data. We train the proposed 3D atrous ConvLSTM network with focal loss to tackle the class imbalance problem commonly seen in background subtraction. Experimental results on a wide range of videos demonstrate the effectiveness of our approach and its superiority over existing methods.

1.2.2 Generative Adversarial Networks with Action Control

We propose new generative adversarial networks (VPGAN) for stochastic video prediction. Our framework, called VPGAN, employs an adversarial inference model

and a cycle-consistency loss function to empower the framework to obtain more accurate predictions. In addition, we incorporate a conformal mapping network structure into VPGAN to enable action control for generating desirable future frames. In this way, VPGAN is able to produce fake videos of an object moving along a specific direction. Experimental results show that the combination of VPGAN with a pre-trained image segmentation model outperforms existing stochastic video prediction methods.

1.2.3 Dual attention generative adversarial networks

We propose a new deep learning approach, specifically a generative adversarial network (GAN) model, to combine SOHO/MDI LOS magnetograms with Hα observations from the Big Bear Solar Observatory (BBSO) for synthesizing B_x and B_y transverse fields. The generated magnetic field components B'_x and B'_y along with the MDI LOS magnetograms, which can be treated as B_z components, create vector magnetograms for different solar cycles. This way we are able to expand the availability of vector magnetograms in different solar cycles that had stronger flare activity. Experimental results show that the generated B'_x and B'_y have good quality, with the average Pearson product-moment correlation coefficient (PPMCC) being approximately 80% based on our test datasets.

1.3 Organization

The structure of this dissertation is organized as follows,

Chapter 2. Three-Dimensional Atrous Convolutional Long Short-Term Memory Network for Background Subtraction

In this chapter, we give a brief picture of the problem of background subtraction and explain our work in details. Experimental results with quantitative data and visualization are also presented.

Chapter 3. Generative Adversarial Networks for Stochastic Video Prediction with Action Control

In this chapter, we introduce the difficulties of video understanding and prediction. Then, followed a detail illustration of our research methodology and experimental results.

Chapter 4. Deep learning-based synthesis of vector magnetograms using generative adversarial networks

In this chapter, we introduce the intuition and astronomical background of the magnetograms prediction problem. A detail description of our research and quantitative experimental results are also presented in this chapter.

Chapter 5. Conclusion

This chapter gives the conclusion part of the dissertation. In this part we have to explain the concluding points which we get during the implementation as well as testing of our research.

Chapter 6. Further Work

This chapter suggests some of the future work to be done with this dissertation which could be useful for further research in this field.

CHAPTER 2

THREE-DIMENSIONAL ATROUS CONVOLUTIONAL LONG SHORT-TERM MEMORY NETWORK FOR BACKGROUND SUBTRACTION

2.1 Background

Separating moving foreground objects from stationary background images in a video sequence captured by static or moving cameras has been an important problem in computer vision [67]. It finds many applications such as video surveillance, traffic monitoring and motion detection. A useful technique for tackling this problem is background subtraction. Background subtraction, also known as foreground detection, is a binary classification task, which designates each pixel in the video sequence as belonging to either the background or foreground scene [12, 3].

This task has been extensively studied in the past. An earlier method was to use Gaussian mixture models (GMMs) to construct the background model and designate each pixel as foreground or background [91]. Probability distributions were adaptively updated using an EM algorithm. Several improved GMMs were later proposed to enhance classification accuracy [36, 53, 107].

Robust principal component analysis (RPCA) [99] was another popular approach for background subtraction. It was based on [71], which considered video frames as a combination of backgrounds, foregrounds and noise. Several variations of the RPCA method were adopted to construct the background model. For example, Zhou et al. [105] developed Decolor, which employed RPCA and a Markov random field (MRF) for foreground detection. Feng et al. [26] introduced online-RPCA (ORPCA); Shakeri and Zhang [83] used ORPCA and Decolor together. There have also been fuzzy [102, 24, 85] and statistical methods [8, 90] for background subtraction. Several

hybrid approaches, such as IUTIS-3 and IUTIS-5 [10], were proposed to combine these methods and run a genetic algorithm on them to select the best combination of the methods. Braham et al. [11] presented a semantic background subtraction method to reduce false positives.

One weakness of the above traditional methods is that they are unable to learn latent feature representations of the background model, and hence are insufficient to handle real-world situations. Recently, some researchers approached the background subtraction problem using deep learning, as deep learning achieved good results in many areas [73, 74, 75, 87]. For example, Braham and Van Droogenbroeck [12], and later Babaee et al. [3], proposed convolutional neural networks (CNNs) [49] to solve the background subtraction problem. They adopted a fixed background model, which was generated from a temporal median operation over N video frames. A scene-specific CNN was trained with corresponding image patches from the background images, video frames and ground truth pixels. Then, a patch around each pixel was fed to the CNN and a score was calculated, which determined the label of the pixel. Wang et al. [97] proposed a cascade CNN model, which was based on a multi-scale convolutional neural network. It used several video frames as input, and the data were run through networks of different scales. The results were then combined together to make predictions. These authors reported that deep learning yields a better result than the traditional methods, as deep learning can capture latent feature representations from the background model.

However, there still exist insufficiencies in the existing deep learning systems. First, some of these systems [12, 97] are scene-specific; they require training samples from specific evaluation data, which may not be available in certain situations. Second, some of the systems [3] only use a small patch around each pixel as input to the CNN, without taking into consideration the whole frame. Third, many of the existing systems [12, 3, 97] require pre- or post-processing of the data, and hence are

not based on an end-to-end learning framework. These existing systems often use 10 frames as input to their CNN model; none of them consider long-term dependencies of the input video sequences.

In this research, we propose a new approach for background subtraction to address the above insufficiencies. The major contributions of our work include: Firstly, a new 3D atrous convolutional neural network (CNN) able to learn deep spatial-temporal features without losing resolution information;

Secondly, a new deep learning model combining our 3D atrous CNN with two convolutional long short-term memory (ConvLSTM) networks.

Finally, training the new deep learning model, named a two-level 3D atrous ConvLSTM network, with focal loss [55] to tackle the class imbalance problem in which the number of pixels belonging to the background scene is much larger than the number of pixels belonging to the foreground scene.

By combining our 3D atrous CNN with two ConvLSTM networks, we can learn both short-term and long-term spatial-temporal information of the input video data. Furthermore, we employ a completely end-to-end framework that doesn't require any pre- or post-processing of the data. Our experimental results show that the proposed deep learning model is more accurate than existing methods.

The rest of this research is organized as follows. Section 2.2 formalizes the problem studied here and details our approach for tackling this problem. Section 2.3 reports experimental results, evaluating the performance of our approach and comparing it with existing methods. Section 2.4 concludes the research and points out some directions for future research.

2.2 Proposed Approach

2.2.1 Problem Formulation

The background subtraction problem addressed in this research is formally defined as follows. Given is a sequence of frames or images $(F_1, F_2, \ldots, F_n)$ of resolution $N \times M$, where each frame F_k, $1 \leq k \leq n$, is a matrix of N columns and M rows of pixels. Our goal is to derive a sequence of binary masks $(Y_1, Y_2, \ldots, Y_n)$ of resolution $N \times M$, such that $Y_k(i, j) = -1$, $1 \leq k \leq n$, $1 \leq i \leq N$, $1 \leq j \leq M$, indicates the corresponding pixel in $F_k(i, j)$ belongs to the background scene and $Y_k(i, j) = 1$ indicates the pixel in $F_k(i, j)$ belongs to the foreground scene.

2.2.2 Three-Dimensional Atrous CNN

Three-Dimensional Atrous Convolution In order to capture both the spatial and temporal information in a video sequence without losing resolution, we propose to use a technique called *3D atrous convolution*. This technique is an enhancement of two previous methods: 3D CNNs [43] and atrous convolution [16].

Two-Dimensional convolution has been frequently used in image processing. It is powerful in extracting spatial information from an image. However, in computer vision and video processing, 2D convolution is insufficient as it does not consider the temporal information encoded in multiple contiguous frames. To effectively incorporate the temporal information into video processing, Ji et al. [43] proposed 3D CNNs and applied them to human action recognition.

On the other hand, while the encoder-decoder model [61] is effective in image segmentation, the multiple combination of max-pooling layers significantly reduces the spatial resolution of the resulting feature maps, causing the loss of spatial information about an image. Even though transposed convolution [61, 101] can recover the spatial resolution, it cannot restore the lost information about the image. To solve this problem, Chen et al. [16] used atrous convolution for image

segmentation without losing spatial resolution where the term "atrous convolution" means convolution with upsampled filters.

The reason why we need atrous convolution in the time domain, besides the spatial domain, is that we want to have a larger view from the time domain. Consider movements that are not significant in just a few frames but become significant in a larger range of frames. Under this circumstance, a larger view from the time domain can help us identify those movements. Traditional convolution with a huge kernel is unrealistic here as it requires a large number of parameters. Therefore, we propose to use 3D atrous convolution in our network, which works as follows.

We define the value at (x, y, z) on the jth feature map in the ith layer, denoted v_{ij}^{xyz}, given kernel size (P_i, Q_i, R_i) of atrous rate (s, t), as follows:

$$v_{ij}^{xyz} = ReLU(b_{ij} + \Omega), \tag{2.1}$$

where

$$\Omega = \sum_{m} \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p\times s)(y+q\times s)(z+r\times t)}. \tag{2.2}$$

Here, $ReLU$ is the rectifier function, b_{ij} stands for the bias for this jth feature map, m indexes over all the feature maps in the $(i-1)$th layer connected to this jth feature map, w_{ijm}^{pqr} is the weight at the position (p, q, r) of the kernel connected to this jth feature map, and P_i, Q_i, R_i are the height, width and length of the kernel, respectively. The atrous rate (s, t) in Equation (2.2) corresponds to the stride with which we sample the input signal; this sampling is equivalent to convolving the input with upsampled filters produced by inserting $s-1$ zeros and $t-1$ zeros between two consecutive filter values along each spatial dimension and temporal dimension respectively, as illustrated in Figure 2.1.

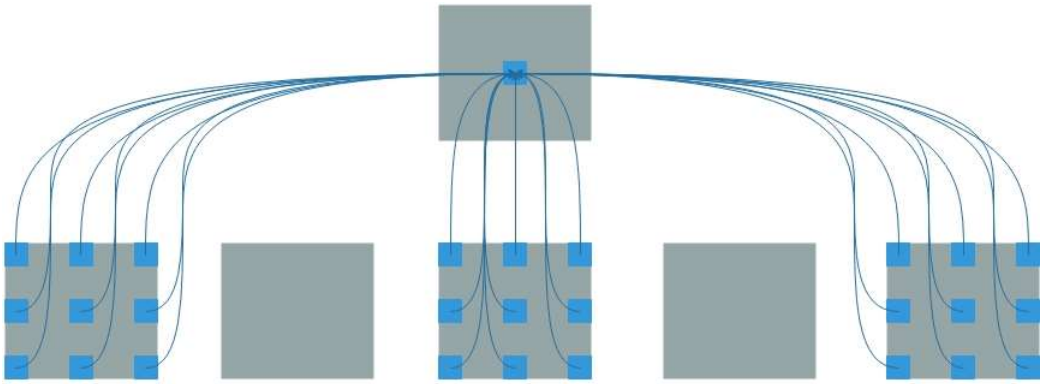

Illustration of Three-Dimensional atrous convolution with kernel size (3,3,3) and rate (2,2).

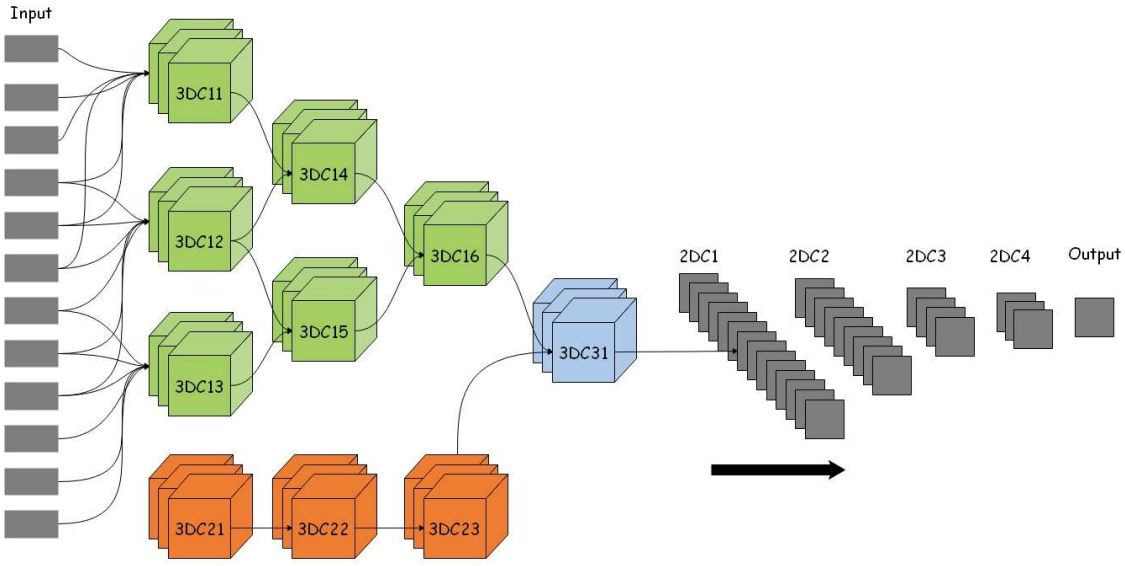

Figure 2.1 Architecture of our feature extractor CNN, which has 10 layers. Layer 1 is the input layer. There are two parallel structures in layers 2, 3, 4 represented by green and orange cubes respectively to gain different temporal information. Their outputs are concatenated in 3DC31 in layer 5. 2D atrous convolution is applied to the remaining layers 6, 7, 8, 9 to eliminate the time dimension and perform image segmentation. Layer 10 is the output layer.

In Figure 2.1, the kernel size is $(3, 3, 3)$, which is represented by 3 blue rectangles at top, 3 blue rectangles in the middle and 3 blue rectangles at bottom in each of the 1st, 3rd, and 5th gray rectangles in the bottom row of Figure 2.1. The atrous rate $(s, t) = (2, , 2)$ is shown as follows. The space between two blue rectangles inside each of the 1st, 3rd, and 5th gray rectangles in the bottom row of Figure 2.1 represents that $s - 1 = 2 - 1 = 1$ zero is inserted between two consecutive filter values along each spatial dimension. The 2nd and 4th gray rectangles in the bottom row of Figure 2.1 represent that $t - 1 = 2 - 1 = 1$ zero is inserted between two consecutive filter values along each temporal dimension.

The CNN Architecture Figure 2.2 illustrates our proposed 3D atrous CNN architecture. This CNN consists of 10 layers, with the first and the last as the input and output layer. For each time step t, the CNN takes as input a sequence of $2k$ frames $(F_{t-k}, \ldots, F_{t+k-1})$ where k is set to 6. In order to have different temporal views of our data, we construct two parallel structures S_1, S_2 in layers 2, 3, 4, represented by green and orange cubes respectively in Figure 2.2.

The first layer of the first structure S_1 focuses on local temporal information instead of the whole sequence of frames. We group six frames together with a stride of 3 and connect them to 3DC11, 3DC12, 3DC13 respectively. Here 3DC11 (3DC12, 3DC13, respectively) represents 3D atrous convolution in unit 1 (2, 3, respectively) of structure S_1. Then 3DC11, 3DC12 are connected to 3DC14, and 3DC12, 3DC13 are connected to 3DC15. Here 3DC14 (3DC15, respectively) represents 3D atrous convolution in unit 4 (5, respectively) of structure S_1. Finally 3DC14, 3DC15 are both connected to 3DC16, where 3DC16 represents 3D atrous convolution in unit 6 of structure S_1. We apply a temporal atrous rate of 1 to the first structure S_1.

The second structure S_2 is comprised of 3DC21, 3DC22, 3DC23 where 3DC21 (3DC22, 3DC23, respectively) represents 3D atrous convolution in unit 1 (2, 3,

respectively) of structure S_2. We group 12 frames together and connect these 12 frames to 3DC21. (For clarity, lines connecting the 12 input frames to 3DC21 are not shown in Figure 2.2.) 3DC21 is then connected to 3DC22, which is connected to 3DC23. In order to conduct a 'skip' view in our model, we apply a temporal atrous rate of 4 to the second structure S_2.

The two parallel structures S_1, S_2 are connected to 3DC31. We then apply 2D atrous convolution to eliminate the temporal dimension, and derive four 2D atrous convolution layers: 2DC1, 2DC2, 2DC3, 2DC4. Finally, the output is the probability mask of frame F_t showing the probability of each pixel in the frame belonging to the foreground scene. In our model, we employ a dropout layer in every 2D atrous convolution layer, with a dropout rate of 0.3 to avoid overfitting.

The novelty of our proposed 3D atrous CNN architecture is that, in contrast to existing CNNs (e.g., [43]), we apply two structures (streams) to the temporal dimension for generating feature maps. Our experimental results indicate that this architecture performs better than architectures using only one stream, S_1 or S_2, for feature map generation.

2.2.3 Our Deep Learning Model

ConvLSTM Network In the past several years, the long short-term memory (LSTM) model, which is a special recurrent neural network (RNN) [66], has been applied to sequence modeling, especially in learning long-term dependencies for speech recognition [34], text generation [93], video processing [22], among others. LSTMs give solutions to the problems of vanishing and exploding gradients by incorporating memory states that enable the networks to learn whether to forget or update hidden states given new information.

In general, an LSTM unit includes a forget gate f, input gate i, output gate o, input modulation gate g, and memory cell c. A drawback of LSTMs is that it is

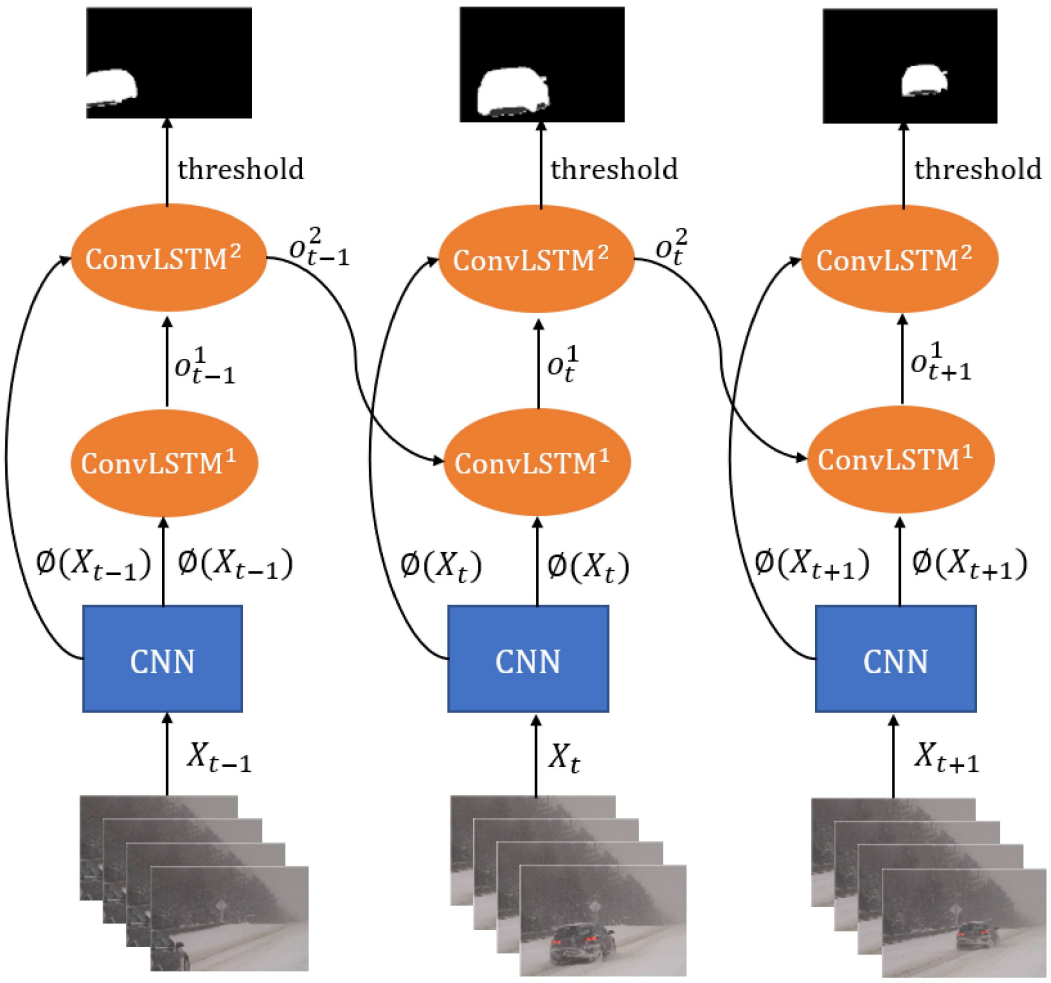

Figure 2.2 Illustration of how the proposed 2-level 3D atrous ConvLSTM network works at time steps $t-1$, t and $t+1$. The input of ConvLSTM^1 at time step t comprises the output of our feature extractor CNN (i.e., $\phi(X_t)$) and the output of ConvLSTM^2 for time step $t-1$ (i.e., o^2_{t-1}). The input of ConvLSTM^2 at time step t comprises the output of our feature extractor CNN (i.e., $\phi(X_t)$) and the output of ConvLSTM^1 (i.e., o^1_t). The input of our feature extractor CNN comprises 12 frames as shown in Figure 2.2; for clarity, only four frames are drawn here.

insufficient to capture spatial information. Shi et al. [84] developed convolutional LSTM (ConvLSTM) models to overcome this problem.

The ConvLSTM updates for time step t are as follows [84]:

$$\begin{aligned} f_t &= \sigma(W_{xf} * x_t + W_{hf} * h_{t-1} + W_{cf} \circ c_{t-1} + b_f), \\ i_t &= \sigma(W_{xi} * x_t + W_{hi} * h_{t-1} + W_{ci} \circ c_{t-1} + b_i), \\ g_t &= \tanh(W_{xc} * x_t + W_{hc} * h_{t-1} + b_c), \\ c_t &= f_t \circ c_{t-1} + i_t \circ g_t, \\ o_t &= \sigma(W_{xo} * x_t + W_{ho} * h_{t-1} + W_{co} \circ c_t + b_o), \\ h_t &= o_t \circ \tanh(c_t), \end{aligned} \tag{2.3}$$

where σ is the logistic sigmoid function, $*$ denotes the convolution operator, $\circ$ denotes the Hadamard product, x_t is the input vector, f_t is the forget gate's activation vector, i_t is the input gate's activation vector, g_t is the input modulation gate's activation vector, c_t is the cell state vector, o_t is the output gate's activation vector, and h_t is the hidden vector. The weight matrix subscripts are self-explanatory; for instance W_{hi} is the hidden-input gate matrix. All the vectors and gates including $x_t, c_t, h_t, f_t, i_t, o_t$ are 3D tensors whose last two dimensions are spatial dimensions. ConvLSTMs can be adopted as building blocks for more complex neural network structures. Multiple levels can be created by using the output gate's activation vector of the ConvLSTM in level $j-1$ as the input to the ConvLSTM in level j.

Two-Level Three-Dimensional Atrous ConvLSTM Network Our deep learning model is constructed by stacking layers of ConvLSTMs above our feature extractor CNN. The rationale behind this design is that, although the 3D atrous CNN can capture deep spatial-temporal features and achieve good performance in background

segmentation, CNNs in general only handle a short sequence of data. For example, Braham and Van Droogenbroeck [12] used 1 frame as input to their CNN networks to perform image segmentation. If the input of these CNN networks is a long sequence of data, the networks would not only suffer from too many parameters (which would consume too much memory), requiring long training and execution time, but also produce poor segmentation results.

To overcome these problems, we propose a new deep learning model, named a two-level 3D atrous ConvLSTM network, by combining our 3D atrous CNN with ConvLSTMs where the 3D atrous CNN is effective in extracting spatial-temporal features without losing resolution information and ConvLSTMs are powerful in manipulating video sequences. Specifically, our model works by passing a sequence of input frames $X_t = (F_{t-k}, ..., F_{t+k-1})$ through a pretrained feature extractor ϕ, which is our proposed 3D atrous CNN shown in Figure 2.2, to produce a fixed-size tensor $\phi(X_t)$. Here $\phi(X_t)$ represents the output tensor of the 2DC4 layer of our 3D atrous CNN (see Figure 2.2). This computed tensor $\phi(X_t)$ then becomes part of the input of our ConvLSTMs. Multiple ConvLSTMs could be stacked on top of the feature extractor ϕ.

Algorithm 1 2-Level 3D Atrous ConvLSTM Network

Input: A sequence of frames $(F_1, F_2, \ldots, F_n)$.
Output: A sequence of binary masks $(Y_1, Y_2, \ldots, Y_n)$.

1: **for** each time step t, $1 \leq t \leq n$, **do**
2: Let X_t be the sequence of $2k$ frames $(F_{t-k}, \ldots, F_{t+k-1})$;
3: Our 3D atrous CNN takes X_t as input and produces $\phi(X_t)$ as output;
4: ConvLSTM^1 takes $x_t^1 = \phi(X_t) \oplus o_{t-1}^2$ as input and updates $f_t^1, h_t^1, i_t^1, o_t^1, c_t^1$ using Equation (2.3);
5: ConvLSTM^2 takes $x_t^2 = \phi(X_t) \oplus o_t^1$ as input and updates $f_t^2, h_t^2, i_t^2, o_t^2, c_t^2$ using Equation (2.3);
6: Let $Y_t = \text{threshold}_{\{1,-1\}}(o_t^2)$;
7: **end for**
8: **return** Y_t, $1 \leq t \leq n$

Figure 2.3 illustrates how our 2-level 3D atrous ConvLSTM network works; Algorithm 7 presents details. For each time step t, $f_t^j, h_t^j, i_t^j, o_t^j, c_t^j$, $1 \leq j \leq 2$,

represent the states of the ConvLSTM in the jth level, denoted ConvLSTM^j. These states are updated using Equation (2.3). The input of ConvLSTM^1 (ConvLSTM^2, respectively) at time step t, denoted x_t^1 (x_t^2, respectively), is the concatenation of the two vectors $\phi(X_t)$ and o_{t-1}^2 ($\phi(X_t)$ and o_t^1, respectively), denoted $\phi(X_t) \oplus o_{t-1}^2$ ($\phi(X_t) \oplus o_t^1$, respectively). The algorithm compares each value in o_t^2, which is the output gate's activation vector of ConvLSTM^2, with a user-determined threshold θ. (In this study, θ is set to 0.5.) If the value is greater than or equal to θ, then the value is replaced by 1; otherwise the value is replaced by -1. This yields a matrix Y_t of 1s and -1s, where Y_t is the binary mask produced by our algorithm for frame F_t at time step t.

The novelty of the proposed deep learning model is that, unlike existing work [22] which combines a CNN with LSTMs, our model combines the proposed 3D atrous CNN with ConvLSTMs. Furthermore, we design a novel way of connections between the 3D atrous CNN and ConvLSTMs which is effective for the background subtraction task.

2.2.4 Training with Focal Loss

Focal Loss The background subtraction problem tackled here is an imbalanced classification problem in which pixels belonging to the background scene outnumber pixels belonging to the foreground scene. Many of the background pixels are very 'easy' to classify, which means that their impact on the loss function is small. However, when summed over a huge number of such easy samples, the small loss values can overwhelm the rare class (foreground). Following Lin et al. [55], we apply the focal loss to our training procedure to handle the class imbalance problem. Let $y \in \{1, -1\}$ specify the ground-truth class and let p represent our model's estimated probability

for the class (foreground) with label $y = 1$. We define q as follows:

$$q = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise.} \end{cases} \tag{2.4}$$

As in [55], we incorporate a modulating factor $(1 - q)^\gamma$ to the cross entropy loss $-\log(q)$ with focusing parameter $\gamma \geq 0$. Let $\alpha \in [0, 1]$ be a weighting factor for class 1 (foreground) and $1 - \alpha$ for class -1 (background). We adopt an α-balanced variant of the focal loss:

$$\mathrm{FL}(q) = -\alpha(1 - q)^\gamma \log(q). \tag{2.5}$$

This is useful in our training process, particularly when we apply the focal loss to our 3D atrous CNN and ConvLSTMs with $\gamma = 2$ and $\alpha = 0.25$.

Training As shown in Algorithm 1, at each time step t, our 2-level 3D atrous ConvLSTM network takes as input a sequence of $2k$ frames $X_t = (F_{t-k}, \ldots, F_{t+k-1})$. We choose $k = 6$ because 12 frames are sufficient to model short-term movement and meanwhile do not require too many parameters for the network. In dealing with videos of different resolutions, we resize the videos to a uniform resolution. Each video is a sequence of frames. We use the first eighty percent of the frames as training data and the last twenty percent of the frames as test data. The pixel values in each frame are normalized into decimal numbers between 0 and 1. We pretrain our 3D atrous CNN using the training data and embed the pretrained CNN into the ConvLSTM models and train the entire 2-level 3D atrous ConvLSTM network again. The parameters of

the pretrained CNN are tunable in our full network. Fine tunning these parameters helps improve our results.

2.3 Experiments and Results

In this section, we describe implementation details of our proposed 2-level 3D atrous ConvLSTM network architecture. A series of experiments were conducted to evaluate the performance of the network architecture and compare it with existing methods. We first introduce the dataset used in the experiments, then show the configurations of the network architecture, define performance metrics and finally present experimental results.

2.3.1 Dataset

We used CDnet2014 [96] in our experiments. CDnet2014 is a video dataset for testing change detection algorithms. This dataset contains eleven video categories with 4 to 6 video sequences in each category. These eleven video categories are: Baseline, Dynamic Background, Shadow, Bad Weather, Low Frame Rate, Turbulence, Night Videos, PTZ, Camera Jitter, Intermittent Object Motion, and Thermal. The resolutions of these videos range from 320×240 to 720×486. We resized the videos to a uniform resolution, namely 320×240.

2.3.2 Configuration

Our deep learning model was implemented using TensorFlow. In training this model, we adopted the RMSProp optimizer [1], with learning_rate = 10^{-6}, decay = 0.9, momentum = 0.0, and epsilon = 10^{-8}. A batch size of 7 was used for training. The focal loss was used as the loss function of our model. The model contains our 3D atrous CNN with 10 layers, and two ConvLSTMs stacked on top of the 3D atrous CNN (see Figures 2.2 and 2.3). Table 2.1 presents details of our network architecture. In the table, for each layer with 3D as a prefix, the first and second numbers of Kernel

Table 2.1 Details of proposed Two-Level Three-Dimensional Atrous ConvLSTM Network Architecture

Layer	Dimension	Kernel	Atrous Rate	Channels
3DC11	3D	(3,3,6)	(1,1,1)	64
3DC12	3D	(3,3,6)	(1,1,1)	64
3DC13	3D	(3,3,6)	(1,1,1)	64
3DC14	3D	(3,3,4)	(2,2,1)	128
3DC15	3D	(3,3,4)	(2,2,1)	128
3DC16	3D	(3,3,12)	(4,4,1)	256
3DC21	3D	(3,3,3)	(1,1,4)	64
3DC22	3D	(3,3,6)	(2,2,2)	128
3DC23	3D	(3,3,6)	(4,4,2)	256
3DC31	3D	(3,3,36)	(1,1,1)	256
2DC1	2D	(3,3)	(1,1)	512
2DC2	2D	(3,3)	(2,2)	256
2DC3	2D	(3,3)	(1,1)	128
2DC4	2D	(3,3)	(1,1)	64
ConvLSTM^1	2D	(3,3)	(1,1)	1
ConvLSTM^2	2D	(3,3)	(1,1)	1

and Atrous Rate represent spatial dimensions, and the third number represents the temporal dimension.

Notice that in our architecture, we connected 2DC4 to ConvLSTM^1 (see Figures 2.2 and 2.3). This default model is denoted as M_{2DC4}. There are other options such as connecting 2DC3 (2DC2 and 2DC1, respectively) to ConvLSTM^1, denoted M_{2DC3} (M_{2DC2} and M_{2DC1}, respectively). Another model, denoted M_{3D}, is only using the pretrained 3D atrous CNN as the learning model without including any ConvLSTM. As our experimental results show later, M_{2DC4} achieves the best performance.

2.3.3 Evaluation

We define a true positive (true negative, false positive, false negative, respectively) to be a pixel in the test data where our model predicts the pixel belongs to the foreground (background, foreground, background, respectively) scene and the pixel is, according to the ground truth, actually in the foreground (background, background, foreground, respectively) scene. TP (TN, FP, FN, respectively) denotes the number of true positives (true negatives, false positives, false negatives, respectively). We

Table 2.2 Performance Evaluation of Our Model on the CDnet2014 Dataset

Category	TPR	TNR	FPR	FNR	PWC	PRE	F_1
Baseline	0.9863	0.9998	0.0003	0.0183	0.0977	0.9937	0.9897
Dynamic Background	0.9831	0.9996	0.0009	0.0179	0.1213	0.9793	0.9789
Shadow	0.9819	0.9994	0.0013	0.016	0.1475	0.9843	0.9813
Bad Weather	0.9734	0.9933	0.0008	0.0124	0.1166	0.9658	0.9609
Low Frame Rate	0.9026	0.9996	0.0024	0.0357	0.3466	0.9024	0.8994
Turbulence	0.9538	0.9999	0.0003	0.0362	0.0679	0.9421	0.9488
Night Videos	0.9538	0.9985	0.0010	0.0447	0.3677	0.9626	0.9489
PTZ	0.9639	0.9998	0.0006	0.0679	0.0973	0.9537	0.9582
Camera Jitter	0.9791	0.9997	0.0019	0.0213	0.1556	0.9756	0.9645
Intermittent Motion	0.9677	0.9996	0.0006	0.0244	0.2546	0.9780	0.9637
Thermal	0.9813	0.9997	0.0003	0.0129	0.0980	0.9899	0.9833

Table 2.3 Performance Comparison of Six Background Subtraction Methods

Method	TPR	TNR	FPR	FNR	PWC	PRE	F_1
Our Model	**0.9701**	**0.9991**	**0.00012**	**0.0224**	**0.246**	**0.9661**	**0.9615**
Cascade [97]	0.9506	0.9968	0.0032	0.0494	1.0722	0.8997	0.9209
IUTIS-5 [10]	0.7849	0.9948	0.0052	0.2151	1.1986	0.8087	0.7717
SemanticBGS [11]	0.7890	0.9961	0.0039	0.2110	0.3281	0.8305	0.7892
DeepBS [3]	0.7545	0.9905	0.0095	0.2455	1.992	0.8332	0.7458
SuBSENSE [90]	0.8124	0.9904	0.0096	0.1876	1.678	0.7509	0.7408

use several metrics to evaluate the performance of our model and compare it with existing methods. These performance metrics include the true positive rate (TPR), also known as sensitivity or recall, true negative rate (TNR) or specificity, false positive rate (FPR), false negative rate (FNR), percentage of wrong classification (PWC), precision (PRE) and F_1 measure, which are defined as follows:

$$TPR = \frac{TP}{TP + FN}$$

$$TNR = \frac{TN}{TN + FP}$$

Table 2.4 The F_1 Values of Our Model and Cascade for Each Category of CDnet2014

Method	Baseline	Dynamic Background	Shadow	Bad Weather	Low Frame	Turbulence	Night Videos	PTZ	Camera Jitter	Intermit. Motion	Thermal
Our Model	**0.9897**	**0.9789**	**0.9813**	**0.9609**	**0.8994**	**0.9488**	**0.9489**	**0.9582**	0.9645	**0.9637**	**0.9833**
Cascade	0.9786	0.9658	0.9593	0.9431	0.8370	0.9108	0.8965	0.9168	**0.9758**	0.8505	0.8958

$$FPR = \frac{FP}{TN + FP}$$

$$FNR = \frac{FN}{TP + FN} \tag{2.6}$$

$$PWC = \frac{(FP + FN) \times 100}{TP + TN + FP + FN}$$

$$PRE = \frac{TP}{TP + FP}$$

$$F_1 = \frac{2 \times TPR \times PRE}{TPR + PRE}$$

We applied our model to the eleven video categories in CDnet2014. Table 2.2 presents the results. It can be seen from the table that our model achieves high F_1 values, even for very challenging categories such as Dynamic Background and Camera Jitter. For Night Videos and Low Frame Rate categories, the performance of our model degrades, mainly due to the resolutions and insignificant changes in the foreground scenes of the videos in these two categories. In Turbulence and PTZ videos, there exist inconsistent background scenes throughout the video frames, which have impact on the segmentation (prediction) results.

Next, we compared our model with five methods developed by others, including SemanticBGS [11], Cascade [97], IUTIS-5 [10], DeepBS [3] and SuBSENSE [90]. These are top methods widely used in the community which are representatives of existing deep learning models and typical traditional models. Among these five existing methods, Cascade [97], SemanticBGS [11] and DeepBS [3] are based on deep

learning while IUTIS-5 [10] and SuBSENSE [90] are traditional methods that are not based on deep learning.

Table 2.3 presents the results. Each number in the table represents the average over the eleven categories of CDnet2014. The best value for each performance metric is highlighted in boldface. We note that, among the five existing methods, Cascade [97], which employs a multi-scale CNN to perform background subtraction, is the best. Our model outperforms Cascade in F_1 measure by 4%, and is much better than the other four existing methods.

Table 2.4 shows the F_1 values of our model and Cascade for each category of CDnet2014. The best F_1 value in each category is highlighted in boldface. It can be seen from Table 2.4 that our model outperforms Cascade [97] in every category, except Camera Jitter, of the CDnet2014 dataset.

Figure 2.4 displays some results produced by our model, Cascade [97] and IUTIS-5 [10] where our model and Cascade are based on deep learning while IUTIS-5 is a traditional method not based on deep learning. The first row in Figure 2.4 shows an input frame taken from one of six categories of CDnet2014 including Baseline, Bad Weather, Camera Jitter, Dynamic Background, Intermittent Object Motion, and Low Frame Rate. The second row shows ground truth data. The third row gives the results produced by Cascade [97] based on the input frames in the first row. The fourth row gives the results of IUTIS-5 [10]. The last row presents the results produced by our model.

It can be seen from Figure 2.4 that the results produced by our model are generally closer to the ground truth data, compared to the results of the other two methods. For instance, look at the results for Bad Weather in the second column. The head of the person in the middle of the scene is hardly seen in the result of Cascade displayed in the third row, and her upper body almost disappears in the result of IUTIS-5 displayed in the fourth row. By contrast, in the result of our model

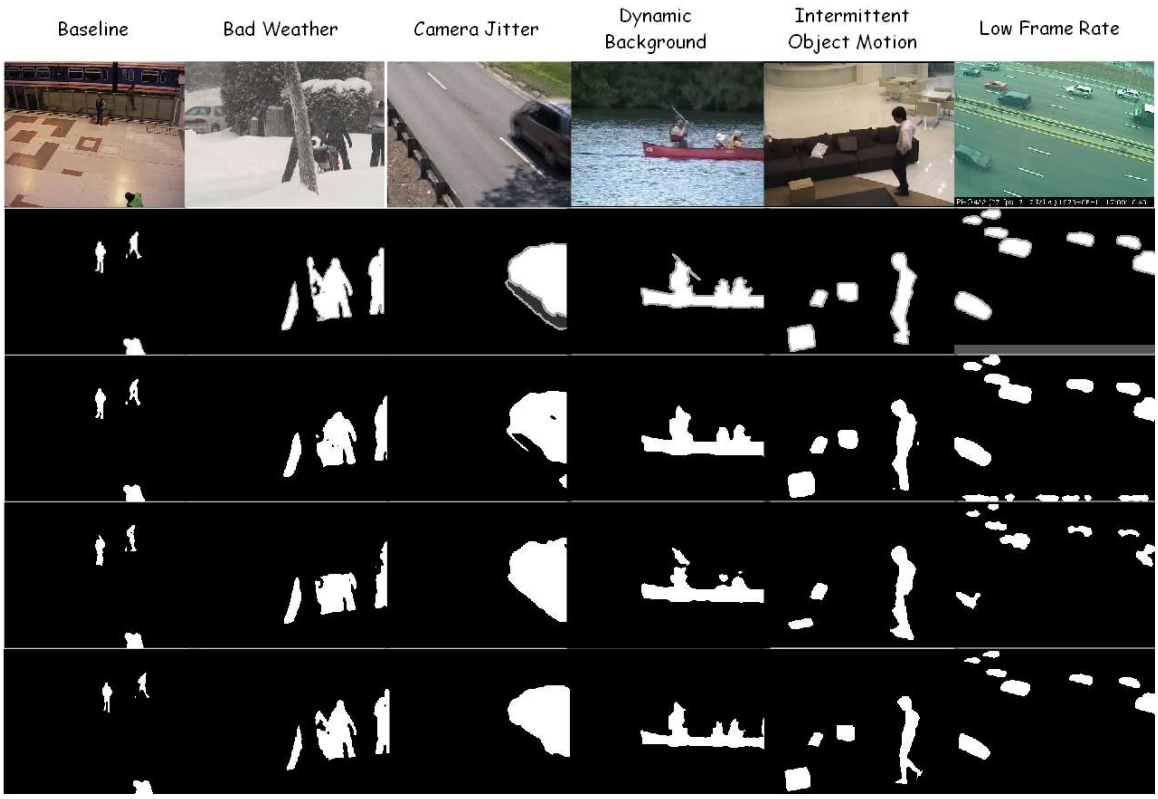

Figure 2.3 Example results produced by Cascade (third row), IUTIS-5 (fourth row) and our model (last row). The first row shows input frames from six categories of CDnet2014. The second row shows the ground truth for each input frame.

displayed in the last row, the head and upper body of this person are clearly shown as in the ground truth.

Finally we compared our default model M_{2DC4} with the other models M_{2DC3} M_{2DC2}, M_{2DC1} and M_{3D} described in Section 3.2. Table 2.5 presents the results. Each F_1 value in the table represents the average over the eleven categories of CDnet2014. The highest F_1 value is highlighted in boldface. Clearly, the M_{2DC4} model, in which the 2DC4 layer of our 3D atrous CNN is connected to ConvLSTM[1], achieves the best performance. This happens probably because the two ConvLSTMs, which serve as the last layers of our pretrained 3D atrous CNN, are both 1-channel 2D convolution layers, and the features from 2DC4 are more suitable for such layers than 2DC3, 2DC2 and 2DC1. The fact that M_{2DC4} beats M_{3D} indicates that stacking two ConvLSTMs on

Table 2.5 The F_1 Values of Five Different Deep Learning Models

Model	F_1
M_{2DC4}	**0.9615**
M_{2DC3}	0.9556
M_{2DC2}	0.9531
M_{2DC1}	0.9512
M_{3D}	0.9521

top of our 3D atrous CNN is effective, allowing us to capture long-term dependencies through the time dimension.

Our 3D atrous CNN accepts as input 12 frames (see Figure 2.2). We have also tested on different numbers of input frames. The 3D atrous CNN uses two streams (structures) S_1 and S_2 for feature map generation, with a temporal atrous rate of 1 for S_1 and a temporal atrous rate of 4 for S_2. We have compared this architecture with alternative architectures using only one structure, S_1 or S_2, or two structures with different temporal atrous rates. Moreover, our proposed deep learning model combines the 3D atrous CNN with two ConvLSTMs (see Figure 2.3). We have tested on alternative models with different numbers of ConvLSTMs. Experimental results showed that the proposed 2-level 3D atrous ConvLSTM network architecture yielded the best performance.

2.4 Summary

In this chapter, we propose a new approach for video background subtraction. Our approach works by combining a 3D atrous convolutional neural network (CNN), used as a feature extractor, with two convolutional long short-term memory (ConvLSTM) networks. This new deep learning model is powerful in processing videos and analyzing sequential data in general. It is not scene specific, and does not require any pre- or post-processing of input video data. We trained this model with focal loss to tackle the class imbalance problem commonly seen in background subtraction. Experimental results on a variety of video sequences demonstrated the effectiveness

of the proposed approach and its superiority over existing methods. In future works we plan to extend the proposed deep learning model for video prediction and segmentation. We are also exploring applications of the model to video processing in scientific domains (e.g., solar physics).

CHAPTER 3

GENERATIVE ADVERSARIAL NETWORKS FOR STOCHASTIC VIDEO PREDICTION WITH ACTION CONTROL

3.1 Background

Acquiring an in-depth understanding of videos has been a cornerstone problem in computer vision. This problem has been studied by various researchers from different perspectives, among which video prediction has attracted much attention. Video prediction aims to generate the pixels of future frames given a sequence of context frames [69, 63]. This task finds many applications ranging from autonomous driving, robotic planning, to object tracking. In practice, unlabeled video sequences can be gathered autonomously from a sensor or recording device. A machine capable of predicting future events using these video sequences in an unsupervised manner will have gained extensive and deep knowledge about its physical environment and surroundings [4, 52].

However, despite its appealing prospects, accurate video prediction remains an open problem. The major challenge is the inherent uncertainty in the dynamics of the world [19]. A typical example is that the future trajectory of a ball hitting the ground is inherently random. Deterministic methods [78, 69, 89] are unable to handle this inherent uncertainty. Furthermore, the improper loss functions adopted in many of the deterministic methods often result in blurry predictions.

With the advent of models such as generative adversarial networks (GANs) [32] and variational auto-encoders (VAEs) [77], the quality of prediction results has been improved. Furthermore, stochastic methods based on these models are able to generate multiple future frames using some randomly sampled noise [4, 19, 52, 38, 54].

However, the adversarial loss functions in GANs tend to be difficult to tune, and these networks suffer from the mode collapse problem, i.e., they select a few prominent modes without being able to adequately cover the space of possible predictions [4, 52]. Moreover, the stochastic methods lack the understanding and control of latent-variable space, rendering action control impossible. Hence, they are unable to generate desirable future frames.

To tackle these problems, we present in this research a new GAN-based framework, named VPGAN, for stochastic video prediction. The main contributions of our work include the following: Firstly, we introduce a new adversarial inference model designed for stochastic video prediction and incorporate a novel cycle-consistency loss into the model to better learn actions that take place in video sequences for enhancing the quality of predicted frames.

Secondly, we incorporate a conformal mapping [2] network structure into our VPGAN framework to enable action control for generating desirable future frames.

Finally, we combine a pre-trained image segmentation model, SegNet [5], with our VPGAN framework to exploit its effectiveness in image understanding. Having more semantic understanding of the frames in video sequences would enable VPGAN to generate more accurate predictions.

To the best of our knowledge, this is the first work to incorporate action control into the video prediction task so as to generate specific future frames. Furthermore, the combination of our VPGAN framework with the pre-trained image segmentation model (SegNet) outperforms existing stochastic video prediction methods as shown in our experimental results reported in the research.

3.2 Related Work

A wide range of deterministic methods for video prediction have been developed in the past. Many of these methods worked by generating future frames from the latent

states of a deep learning model such as recurrent neural networks (RNNs) [66] and convolutional long short-term memory networks (ConvLSTMs) [84]. Ranzato et al. [78] designed an RNN-like network for natural language processing, which was able to predict future frames in a discrete space of patch clusters. Srivastava et al. [89] applied long short-term memory (LSTM) networks to capture long-term pixel dynamics of videos. Oh et al. [69] adapted an encoder-decoder model to encode action movement into the encoder and generate frames conditioned on the action movement. Oliu et al. [70] stacked multiple double-mapping GRU (gated recurrent unit) layers to build a folded recurrent neural network for future video prediction. All these deterministic methods suffered from the inherently blurry predictions obtained from the standard mean squared error (MSE) loss function.

Mathieu et al. [63] developed a stochastic GAN-based model for video prediction. This model used an adversarial loss function instead of least absolute deviations (L_1 loss) and least square errors (L_2 loss). Tulyakov et al. [92] described the MoCoGAN framework, which decomposed a video into a content part and a motion part. They trained two GANs together, one of which was a motion GAN and the other was a content GAN, for video generation. Similar decomposition schemes have been used by Denton and Birodkar [20] and Villegas et al. [94] who developed the MCnet+Res system. These decomposition methods borrow the idea from background subtraction techniques [40] and work well in rather simple scenarios. Hou et al. [38] introduced a bidirectional constraint network, BCnet-D, and used a video adversarial loss function to constrain the motion of predictions. Li et al. [54] solved the video prediction problem in two phases: a multiple time step flow prediction phase followed by a flow-to-frame synthesis phase where the second phase is modeled as a generative process. However, these GAN-based models suffer from problems such as mode collapse and instability [32]. These problems have been addressed by several researchers. For example, Gulrajani et al. [35] proposed to use the Wasserstein

distance to handle training instability. Mescheder et al. [65] provided some training mechanisms and tricks to overcome such problems.

Apart from GAN-based models, another popular technique for generative models is the VAE-based framework [77]. This framework aims to minimize the reconstruction loss and regularization term (KL divergence between a posterior distribution and a prior). It employs a Bayesian support vector machine, permitting efficient Bayesian inference. VAE-based models have acquired great success in image generation [77, 95]. However, they suffer from the same problem when used in video prediction. Specifically, the L_2 reconstruction loss function used by these models tends to produce blurry results as it generates the expected value of all the possibilities for each pixel independently [4, 52]. Hence, few works have applied VAEs directly to video prediction.

The fact that GANs lack Bayesian inference while VAEs suffer from an inappropriate loss function leads to combining GANs with VAEs. State-of-the-art VAE-GAN hybrids include SVG-LP [19] and SV2P [4, 52], both of which perform stochastic video prediction. Notice that VAE-GAN hybrids are not the only type of models that incorporate inference mechanisms into GANs. There are many other efforts such as ALI [23] and BiGANs [21]. In particular, the adversarially learned inference (ALI) model jointly learns a generation network and an inference network using an adversarial process, and has been successfully applied to some semi-supervised learning tasks.

In this work, we propose a new adversarial inference model in the spirit of ALI and designed specifically for video prediction, and incorporate it into our VPGAN framework. This adversarial inference model is totally different from the Bayesian inference used in the VAE-GAN hybrids such as SVG-LP [19] and SV2P [4, 52]. Furthermore, VPGAN employs a cycle-consistency loss function to enhance the quality of prediction results. We show experimentally that the combination of our

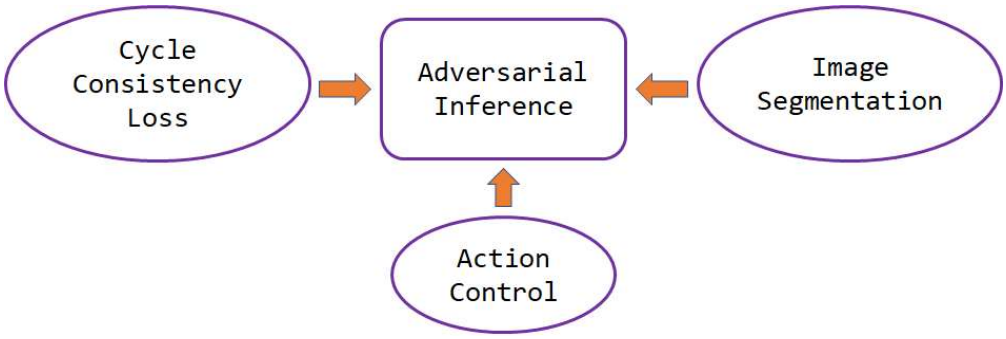

Figure 3.1 Overview of our VPGAN framework. The framework has four components responsible for adversarial inference, cycle-consistency loss, image segmentation and action control respectively.

VPGAN with a pre-trained image segmentation model, SegNet [5], outperforms the existing VAE-GAN hybrids including SVG-LP and SV2P as well as other methods.

The rest of this chapter is organized as follows. Section 3.3 formalizes the problem studied here and details our approach for tackling this problem. Section 3.4 reports experimental results, evaluating the performance of our approach and comparing it with the existing methods. Section 3.5 concludes the work and points out some directions for future research.

3.3 The Proposed Approach

Figure 3.1 presents an overview of our VPGAN framework. The framework consists of (i) the adversarial inference model mentioned in the previous section for stochastic video prediction, (ii) the cycle-consistency loss module that introduces cycle constraints into the prediction space to improve overall accuracy, and (iii) the image segmentation model, SegNet, to further improve prediction accuracy. The action control module aims to generate desirable future frames in which an object moves along a specific direction, and has little impact on prediction accuracy.

3.3.1 Problem Formulation

The task of stochastic video prediction can be formalized as learning a multi value function $f : R^{N\times M\times T} \mapsto R^{N\times M}$ from a collection of T context frames $X_0, \ldots, X_{T-1}$, each of which is a matrix of N rows and M columns of pixels, to some possible future frames $\{X_T\}$.

It is natural to think that the transformation from frame X_{t-1} to frame X_t is caused by some variation Z_t. In [20, 94, 92], the latent variable Z_t is considered as the motion of objects. However, in practice, Z_t contains not only object motion, but also variations of the physical environment and surroundings. In fact, due to adding some constraints to the latent variable, Z_t is the accumulation of multiple factors, i.e., $Z_t = Z_t^1 + Z_t^2 + \ldots + Z_t^k$. Furthermore, because the variation between frames is small as environmental changes usually don't take place in a sudden, we assume that the prior distribution of Z_t is a standard Gaussian $N(\mathbf{0}, \mathbf{I})$. Based on this assumption, the video data can be described as a sequence of pairs $(X_0, Z_0), \ldots, (X_t, Z_t)$, $0 \leq t < T$.

3.3.2 Adversarial Inference

Let X represent the frames and let Z represent the variations under consideration. Let $p_{data}(X)$ represent the true distribution of X. We wish to construct a joint distribution $q(X, Z)$ such that $q(X, Z)$ is a good approximation of $p_{data}(X)$. In practice, it is hard to match $q(X, Z)$ with $p_{data}(X)$. On the other hand, because the video data can be described as a sequence of pairs $(X_0, Z_0), \ldots, (X_t, Z_t)$, we can consider matching $q(X, Z)$ with the joint distribution of (X_t, Z_t), denoted $p(X, Z)$. When $q(X, Z)$ and $p(X, Z)$ are matched, their marginal distributions are also matched. However, performing the matching using a traditional loss such as MSE and L_1 would result in blurry predictions. Instead, we incorporate a new adversarial inference model into our framework. By playing a min-max game between the true evidence (X_t, Z_t) and generated fake sample (X_t', Z_t'), we can match $q(X, Z)$ with $p(X, Z)$.

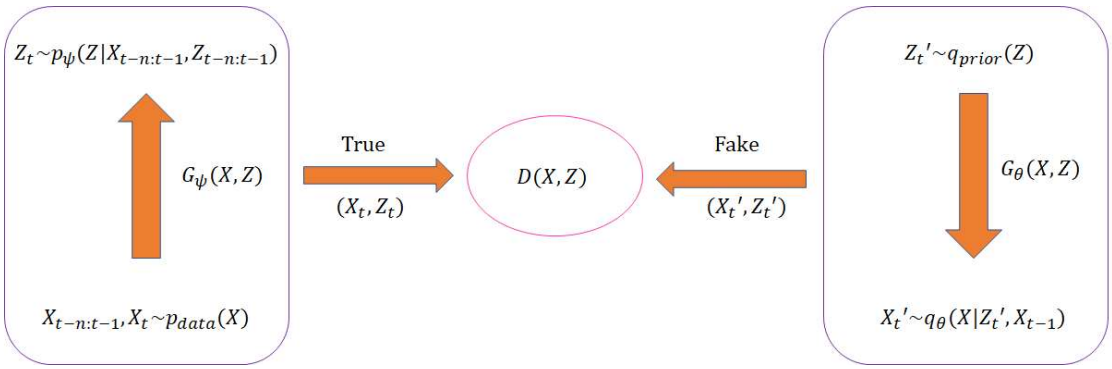

Figure 3.2 Illustration of the VPGAN learning process. Both of G_ψ, G_θ are generators. Discriminator $D(X, Z)$ tries to discriminate between true (X, Z) and fake (X', Z').

Figure. 3.2 illustrates the VPGAN learning process during training. VPGAN employs two generators: $p_\psi = G_\psi(X, Z)$ and $q_\theta = G_\theta(X, Z)$. Let $X_{t-n:t-1}$ denote the frames $X_{t-n}, \ldots, X_{t-1}$ and let $Z_{t-n:t-1}$ denote the variations $Z_{t-n}, \ldots, Z_{t-1}$. Intuitively, past variations should have a 'momentum impact' on the present variation. Thus, we generate the variation at time t, Z_t, conditioned on the past frames $X_{t-n}, \ldots, X_{t-1}$ and past variations $Z_{t-n}, \ldots, Z_{t-1}$. That is to say, $Z_t \sim p_\psi(Z|X_{t-n:t-1}, Z_{t-n:t-1})$. Variations $Z_{t-n:t-1}$ are contained in the frames $X_{t-n:t-1}$ but specifying them explicitly through the input would help p_ψ focus more on the 'momentum impact.' The generator p_ψ in this case could be viewed as an encoder that encodes the past variations $Z_{t-n}, \ldots, Z_{t-1}$ into the latent variable space.

On the other hand, we generate the fake frame at time t, X_t', conditioned on variation Z_t' sampled from a prior, $q_{prior}(Z)$, and a single past frame X_{t-1}, i.e., $X_t' \sim q_\theta(X|Z_t', X_{t-1})$. Here, conditioning on one single past frame is reasonable as Z represents the changes between frames, and conditioning on less information would enforce Z to learn the 'true' variation efficiently. Thus, the generator q_θ serves as a decoder in our framework, which decodes the variation Z_t' and generates new frame X_t'.

The symbol $D(X, Z)$ in Fig. 3.2 represents the discriminator, which tries to distinguish between the true evidence (X_t, Z_t) and the generated fake sample (X'_t, Z'_t). VPGAN keeps on generating fake samples until the discriminator $D(X, Z)$ can not distinguish between the true evidence and generated fake sample, at which moment the training process terminates. When the training is completed, the two joint distributions $q(X, Z)$ and $p(X, Z)$ match with each other.

Denote $p_\psi(Z|X_{t-n:t-1}, Z_{t-n:t-1})$ by $G_\psi(X_{t-n:t-1},$
$Z_{t-n:t-1})$ and $q_\theta(X|Z'_t, X_{t-1})$ by $G_\theta(Z'_t, X_{t-1})$. The adversarial loss function used in the training is calculated as:

$$\begin{aligned} L_{adv} &= E_{X_t \sim p_{data}(X)}[\\ &\quad \log D(X_t, G_\psi(X_{t-n:t-1}, Z_{t-n:t-1}))] \\ &+ E_{Z'_t \sim q_{prior}(Z)}[\\ &\quad 1 - \log D(G_\theta(Z'_t, X_{t-1}), Z'_t)] \end{aligned} \tag{3.1}$$

Fig. 3.3 illustrates the VPGAN feed-forward inference process during testing. The figure shows how to generate or predict the next frame X_t based on the past frames $X_{t-n:t-1}$. First, the past frames $X_{t-n:t-1}$ and past encoded vectors $Z_{t-n:t-1}$ are sent to the encoder p_ψ, which generates the next encoded vector (variation) Z_t. Then the decoder q_θ takes X_{t-1} and Z_t together, and predicts the next frame X_t. Depending on different variations (latent variables) Z_t, q_θ can predict multiple possible next (future) frames $\{X_T\}$.

During training and inference, we calculate p_ψ and q_θ as follows:

$$p_\psi(Z|X_{t-n:t-1}, Z_{t-n:t-1}) \sim N(\mu_\psi(X, Z), \sigma_\psi(X, Z)\mathbf{I}) \tag{3.2}$$

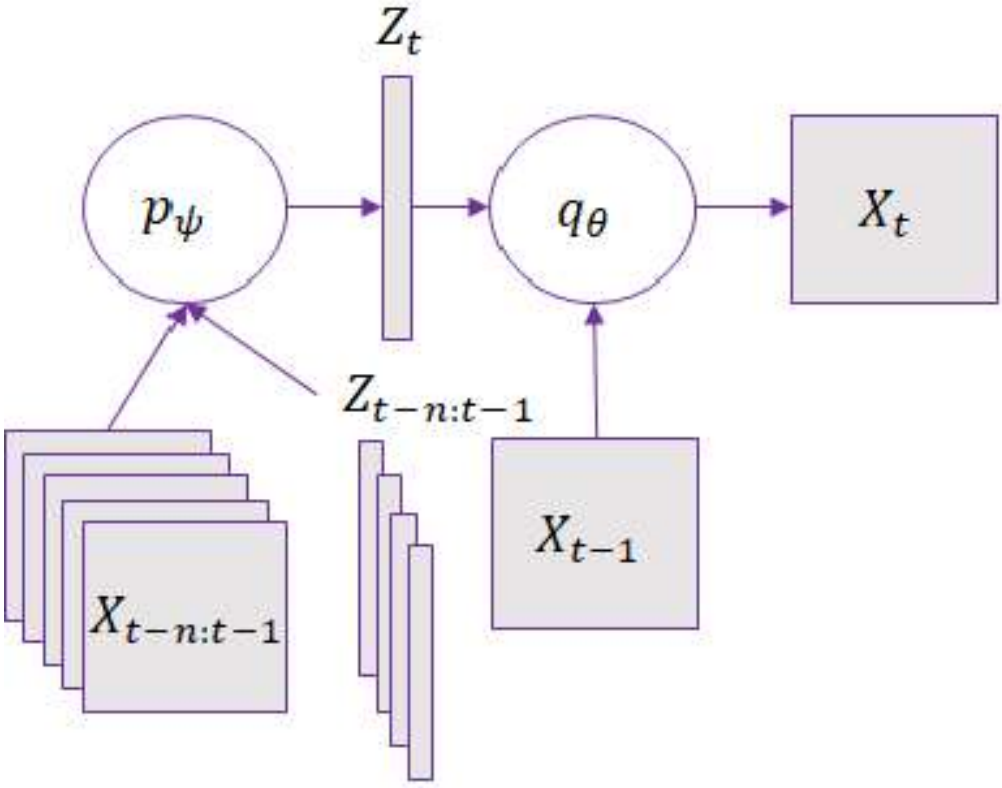

Figure 3.3 Illustration of the VPGAN inference process.

$$q_\theta(X|Z_t, X_{t-1}) \sim N(\mu_\theta(X, Z), \sigma_\theta(X, Z)\mathbf{I}) \tag{3.3}$$

Based on the assumption that the prior distribution of Z is a standard Gaussian, we have

$$q_{prior}(Z) \sim N(\mathbf{0}, \mathbf{I}) \tag{3.4}$$

The sampling procedure used in calculating p_ψ and q_θ can be computed by the reparameterization trick [46]. Specifically, instead of sampling directly from the Gaussian function with the complicated parameters, we treat the sampling procedure as a deterministic transformation of some noise such that the transformation's distribution is computable. Thus, we calculate Z_t as:

$$Z_t = \mu_\psi(X, Z) + \sigma_\psi(X, Z) \odot \xi, \;\; \xi \sim N(\mathbf{0}, \mathbf{I}) \tag{3.5}$$

where $\odot$ denotes the Hadamard product (element-wise product).

3.3.3 Cycle-Consistency Loss

Cycle consistency is based on the idea of using transitivity as a way to regularize structured data. Here we propose a new cycle-consistency loss function for video prediction. With the same generator in (3.3), we generate the frame at time $t-1$, X_{t-1}, conditioned on the opposite of Z_t and X_t. That is, we generate $\bar{X}_{t-1}$ conditioned on $-Z_t$ and X_t where $\bar{X}_{t-1}$ is approximately equal to X_{t-1} as expressed in (3.6) below:

$$X_{t-1} \approx \bar{X}_{t-1} \sim q_\theta(X| - Z_t, X_t) \tag{3.6}$$

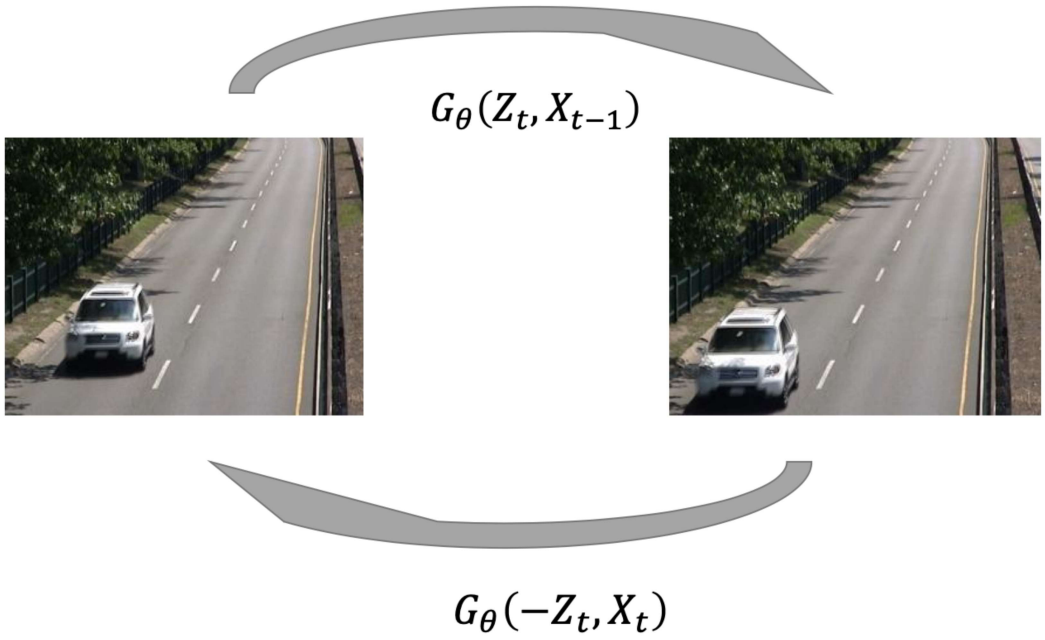

Figure 3.4 Illustration of cycle consistency in our framework.

This is reminiscent of the cycle-consistency loss used for image-to-image translation in [106]. However, our cycle-consistency loss function is different from that in [106] because our loss function is mainly designed for video prediction rather than image translation. Since the prior Z_t follows a standard Gaussian distribution (cf. (3.4)), it is natural to consider the opposite variation to be the negative of Z_t. Fig. 3.4 illustrates how cycle consistency works in our framework. As shown in the figure, we generate the current frame X_t (right) conditioned on the previous frame X_{t-1} (left) and variation Z_t. On the other hand, with the same generator, we generate the previous frame X_{t-1} conditioned on the current frame X_t and the negative of Z_t.

Mathematically, denote $q_\theta(X|Z_t, X_{t-1})$ by $G_\theta(Z_t, X_{t-1})$ and $q_\theta(X| - Z_t, X_t)$ by $G_\theta(-Z_t, X_t)$. Our cycle-consistency loss is calculated as:

$$\begin{aligned} L^1_{cycle} &= E_{X_t, X_{t-1} \sim p_{data}(X)}\{ \\ &\quad \| X_t - G_\theta(Z_t, G_\theta(-Z_t, X_t)) \|_1 \\ &+ \| X_{t-1} - G_\theta(-Z_t, G_\theta(Z_t, X_{t-1})) \|_1\} \end{aligned} \tag{3.7}$$

Here, we utilize L_1 loss as the reconstruction loss. The loss function L^1_{cycle} in (3.7) only considers one-step cycle consistency. We can generalize the formula in (3.7) to take into account cycle consistency of multiple steps (more precisely, k steps) for video prediction. We first define a single-multi loss as follows:

$$\begin{aligned} l^k_{cycle} &= E_{X_t, X_{t-k} \sim p_{data}(X)}\{\| X_t - G_\theta(Z_t, \\ &\quad G_\theta(Z_{t-1}, \ldots, G_\theta(-Z_t, X_t))) \|_1 \\ &+ \| X_{t-k} - G_\theta(-Z_t, \\ &\quad G_\theta(-Z_{t-1}, \ldots, G_\theta(Z_t, X_{t-k}))) \|_1\} \end{aligned} \tag{3.8}$$

Our multi (k steps) cycle-consistency loss is generalized by summing up all single-multi losses as follows:

$$L^k_{cycle} = \sum_{i=1}^{k} a_i \cdot l^i_{cycle} \tag{3.9}$$

It could be very time consuming to calculate L^k_{cycle} for $k \geq 2$ since the procedure includes iterative calculation of generator G_θ. For instance, l^2_{cycle} would require up to 8 times calculation, and the overall L^2_{cycle} would require 12 times calculation.

When using the multi cycle-consistency loss, it is natural to take into account a multi-reconstruction loss. The computational cost is mainly involved in calculating L^k_{cycle}; computing the multi-reconstruction loss takes relatively less time. Define l^k_{recon} as follows:

$$\begin{aligned} l^k_{recon} &= E_{X_t,\ldots,X_{t-k}\sim p_{data}(X)}\{ \\ &\quad \delta(X_t, G_\theta(Z_t, ..., G_\theta(Z_{t-k+1}, X_{t-k})))\} \end{aligned} \tag{3.10}$$

Here, δ stands for the reconstruction distance between the ground truth X_t and $G_\theta(Z_t, ..., G_\theta(Z_{t-k+1}, X_{t-k}))$. Generally, L_2 loss is applied, but in order to obtain more accurate results and reduce the impact of 'future averaging,' we let δ be the L_1 plus perceptual loss [44]. Then, L^k_{recon} is naturally defined as:

$$L^k_{recon} = \sum_{i=1}^{k} b_i \cdot l^i_{recon} \tag{3.11}$$

Combining the multi cycle-consistency loss and multi-reconstruction loss defined in this subsection, our overall loss, denoted L_{loss}, is calculated as follows:

$$L_{loss} = \alpha L_{adv} + \beta L^k_{cycle} + \lambda L^k_{recon} \tag{3.12}$$

The perceptual loss [44] is widely applied in evaluating the reconstruction quality of images. It could be the distance on the Kth feature map, for some K,

of some convolutional neural network, such as VGG16 [86] or ResNet [37] pre-trained on ImageNet [17]. In our study, we applied the simple ResNet [37] to our model.

Notice that, although the multi cycle-consistency loss enforcing long-dependency consistency likely enables more accurate action learning and prediction, its training and inference time would be approximately k times larger than that for the one-step cycle-consistency loss. Moreover, it may suffer from gradient loss. Therefore, in our VPGAN framework, we only utilize the one-step cycle-consistency loss given in (3.7). The evaluation of different k values for L^k_{cycle} will be presented in the experiments section.

It should be pointed out that, in a recent study [38] Hou et al. also incorporated transitivity regularization into their BCnet-D model. The main difference between our VPGAN and BCnet-D is that our backward function $q_\theta(X| - Z_t, X_t)$ in (3.6) and forward function $q_\theta(X|Z_t, X_{t-1})$ in (3.3) are designed in a consistent way and are both optimized during training while Hou et al. used a pre-trained backward function. Our loss function results are thus quite different from those of Hou et al. We generalize our formulas to take into account cycle consistency of multiple steps (k-steps cycle-consistency loss).

3.3.4 Image Segmentation

When constructing the generators G_θ and G_ψ for video prediction, as illustrated in Fig. 3.2, existing methods such as SVG-LP [19] and SV2P [4] employ long short-term memory (LSTM) and convolutional neural networks (CNNs) to capture the spatio-temporal information in the video data. Here we propose to further combine a pre-trained image segmentation model with the generators G_θ and G_ψ, which are implemented by our previously developed convolutional long short-term memory and CNN network [40]. The motivation behind this combination is that when performing the video prediction task, a method needs to understand the scene

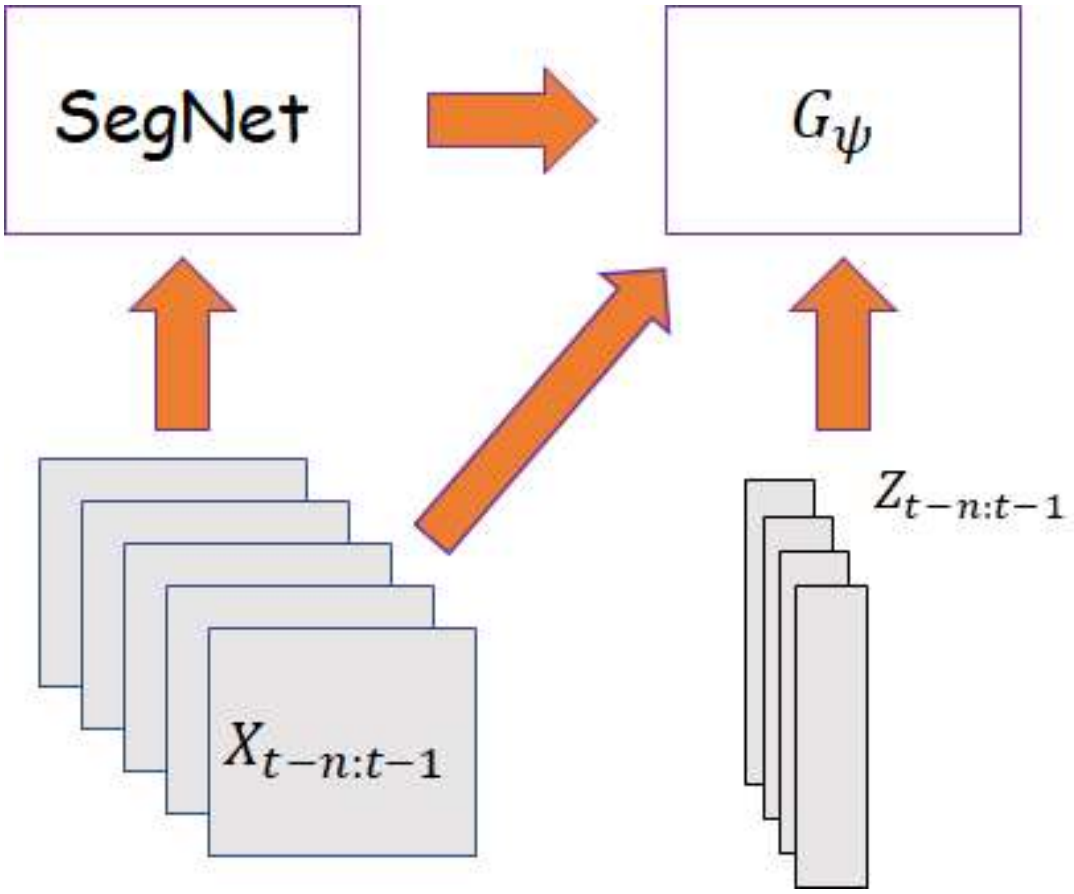

Figure 3.5 Combining the pre-trained image segmentation model SegNet, used as a feature extractor, with the generator G_ψ in VPGAN.

and variations between frames, where the variations between frames are mainly caused by the interactions between objects. Therefore, recognizing the objects and understanding their interactions are critical in predicting the variations.

Since there are well-performed image segmentation models such as SegNet [5], U-Net [79] and DeepLabv3 [15], using one of them as a feature extractor stacked below the generators G_θ and G_ψ would achieve appealing results. For example, Fig. 3.5 shows a combination of SegNet with the generator G_ψ in VPGAN. G_ψ takes, as input, $X_{t-n:t-1}$, $Z_{t-n:t-1}$ and the output of SegNet, and produces, as output, generated variations. In our framework, we choose SegNet due to its better performance than the other image segmentation models as our experimental results show later.

3.3.5 Action Control

In practice, it is natural to consider the generation of desirable images or videos using GANs. Since GANs generally generate data from a random sample of the latent variable space $\mathcal{Z}$, it is hard to control the behavior of GANs. In this subsection, we propose new techniques for generating desirable frames using GANs.

In Section 3.3.3, we use Z_t and $-Z_t$ to represent the opposite variations in the video space $\mathcal{H}$. Specifically, for a movement dataset, Z_t should be able to learn the moving direction of an object, and then $-Z_t$ should mainly represent the object's moving in the opposite direction. That is, from the encoding space (i.e., latent variable space) $\mathcal{Z}$ to the video space $\mathcal{H}$, we preserve what we call a 'symmetry' property, meaning that if Z_1, Z_2 are symmetric in the encoding space $\mathcal{Z}$, then the corresponding generated movements should be symmetric in the video space $\mathcal{H}$.

In addition, we wish to manipulate the latent variable space $\mathcal{Z}$ so as to generate desirable moving directions, through preserving 'orthogonality,' or more precisely, through preserving angles between the encoding space $\mathcal{Z}$ and the moving direction of an object. This orthogonality property can be preserved by first enforcing the latent variable space $\mathcal{Z}$ to be a subset of R^2. Although the moving direction of an object in a video sequence is in R^2, the latent variable $Z \in \mathcal{Z} \subset R^2$ may not simply represent the moving direction of the object, for the following reasons: Firt, the moving direction and Z may not be in the same coordinate system, as the decoder from the latent variable space $\mathcal{Z}$ to the video space $\mathcal{H}$ may contain rotation operations.

Second, the latent variable Z may contain not only direction information, but also velocity, momentum, and other information.

Third, the latent variable Z may contain information related to environmental changes.

Thus, the angles between any two vectors in the latent variable space $\mathcal{Z}$ may not be preserved in the decoding process. To overcome this problem, we add a network to

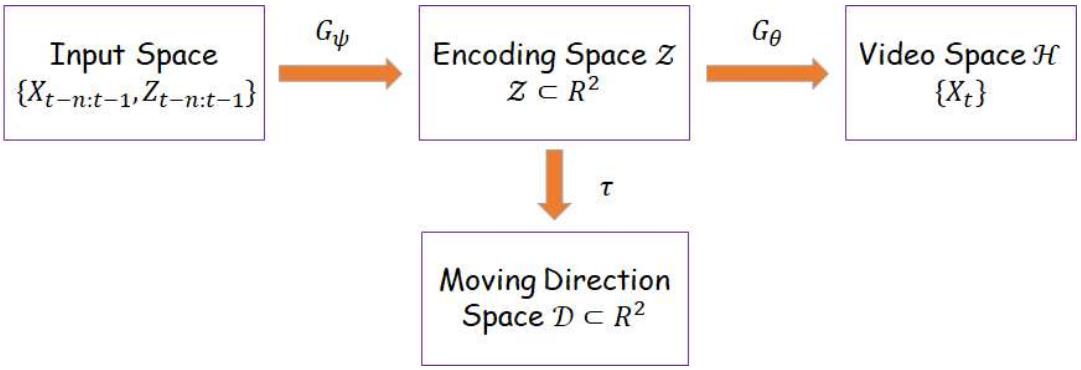

Figure 3.6 Illustration of our modified model for action control.

our framework to preserve such angles. This network acts as a mapping, denoted τ, which maps a latent variable from the latent variable space $\mathcal{Z}$ to the moving direction space $\mathcal{D} \subset R^2$. The moving direction $v(X_{t-1}, X_t)$ of an object between frames X_{t-1} and X_t can be computed by running an optical flow algorithm [9]. Thus, our modified model consists of two decoders: one from the latent variable space $\mathcal{Z} \subset R^2$ to the video space $\mathcal{H}$, and the other decoder, τ, from the latent variable space $\mathcal{Z}$ to the moving direction space $\mathcal{D} \subset R^2$. Fig. 3.6 illustrates this modified model.

The moving direction loss, denoted L_{moving}, is calculated as:

$$L_{moving} = |\frac{< \tau(Z), v(X_{t-1}, X_t) >}{|\tau(Z)| \cdot |v(X_{t-1}, X_t)|} - 1| \tag{3.13}$$

where $< \cdot >$ represents the inner product of two vectors. Such a loss function penalizes the angle difference between two vectors. Our overall training loss is updated to take into account the moving direction loss, and is calculated as:

$$L_{loss} = \alpha L_{adv} + \beta L^k_{cycle} + \lambda L^k_{recon} + \mu L_{moving} \tag{3.14}$$

The Adam optimizer [47] is employed to optimize L_{loss}.

Based on a mathematical concept known as 'conformal mapping' [2], we introduce and add the network, τ, to our model. Formally, a mapping $\mathbf{f} = (f_1, \ldots, f_n)$ where $\mathbf{f} : U \to V$, $U, V \subset R^n$, is conformal (or angle-preserving) at a point $u_0 \in U$ if it preserves the orientation and angles between directed curves through u_0. A mapping $\mathbf{f}$ is conformal iff it is homomorphic and its derivative is nowhere zero, i.e.,

$$\mathbf{J} = \frac{d\mathbf{f}}{d\mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \neq \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \tag{3.15}$$

In our VPGAN framework, the mapping τ is implemented using a 3-layer affine transformation [31, 68]. Such an affine transformation enforces τ to be conformal; therefore it preserves the angle between any two vectors through $'0'$. In this way, if we know a latent variable Z moving toward a specific direction, we can then control the generated moving direction by manipulating the latent variable Z (through rotating with some angle since the angle is preserved between the latent variable space $\mathcal{Z}$ and the moving direction space $\mathcal{D}$). Under this circumstance, we actually do not need to know details concerning Z, such as velocity, momentum and other information. Algorithm 1 depicts our action control procedure.

Algorithm 2 Action Control

1: Sample n sets of continuous frames $\{X_{t-1}, X_t\}_n$ in which objects move toward the same direction.
2: Encode the frames into $Z_t^1, \ldots, Z_t^n$ in the latent variable space $\mathcal{Z}$.
3: Calculate the mean of $Z_t^1, \ldots, Z_t^n$, and denote the mean by $\bar{Z}$.
4: Compute the angle ς between the desired direction and sampled direction.
5: Rotate $\bar{Z}$ by the angle ς in the latent variable space $\mathcal{Z}$.
6: Decode $\bar{Z}$ into the video space $\mathcal{H}$.

The advantages of our proposed action control algorithm are the following:

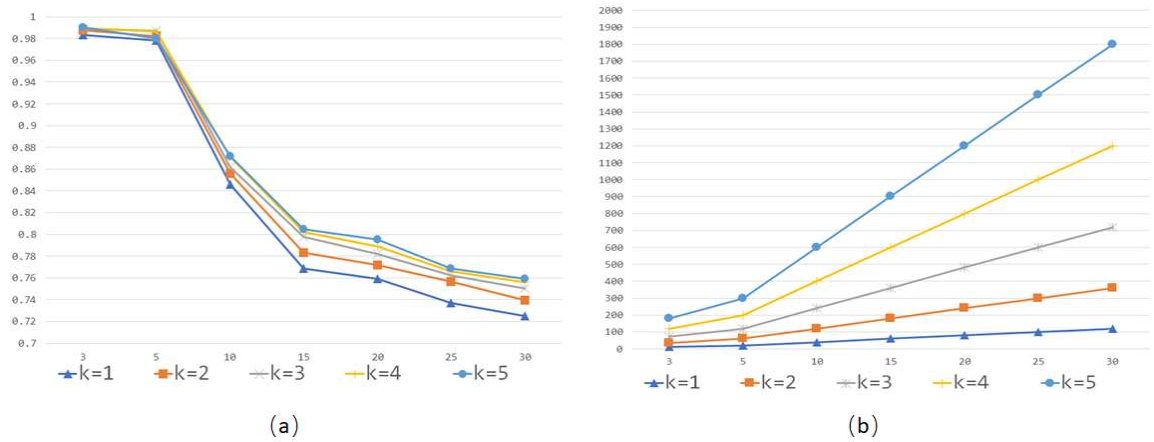

Figure 3.7 Impact of k on the (a) SSIM measure and (b) running time of VPGAN on the Moving Mnist dataset. Given 10 context frames, VPGAN predicts 30 frames recursively, one by one. The X-axis represents different T (time step) values, and for each T value the Y-axis represents the (a) SSIM value computed based on the ground truth and the frame predicted in step T and (b) running time of VPGAN for generating the T frames.

- It suffices to enforce a conformal mapping τ from the latent variable space (i.e., encoding space) $\mathcal{Z}$ to the moving direction space $\mathcal{D}$ (see Fig. 3.6). It is not necessary to handle the latent variables $Z_t^1, \ldots, Z_t^n$ individually.

- Even when the latent variables accumulate many different factors, such as environmental changes, momentum information and so on, our action control algorithm is still able to generate objects moving in the desired direction.

3.4 Experiments and Results

A series of experiments were conducted to evaluate the performance of our VPGAN framework using different datasets, including Moving Mnist [89], BAIR [27], KTH [81] and UCF101 [88]. Moving Mnist [89] is a simple dataset and we used it for generating desired movements. BAIR, KTH and UCF101 are more complicated datasets. We evaluated our cycle-consistency loss function, compared different image segmentation models, and performed ablation studies using these four datasets. To our knowledge, the current best methods on BAIR are SVG-LP [19] and SV2P [4, 52]. The current

best methods on KTH are SVG-LP, SV2P and BCnet-D [38]. The current best methods on UCF101 are BCnet-D and MCnet+Res [94]. We compared our VPGAN with the current best methods on the respective datasets. As in [19, 4, 38, 98], we adopt two performance metrics: structural similarity index measure (SSIM) and peak-signal-to-noise ratio (PSNR). The higher the metric values a method has, the more accurate the method is.

Our experimental methodology is as follows. We used the training data available in each dataset (Moving Mnist, BAIR, KTH) to train VPGAN. Then we took 10 testing frames from each dataset and used the 10 frames as context frames, denoted C_1, C_2, ..., C_{10}. In step 1, the trained VPGAN predicted or generated a frame F_1 at T=1 conditioned on the 10 given context frames C_1, C_2, ..., C_{10}. In step 2, the trained VPGAN generated a frame F_2 at T=2 conditioned on the last 9 given context frames C_2, C_3, ..., C_{10} and the predicted frame F_1. In step 3, the trained VPGAN generated a frame F_3 at T=3 conditioned on the last 8 given context frames C_3, C_4, ..., C_{10} and the two predicted frames F_1 and F_2. We used VPGAN to predict or generate n frames in total (e.g., $n = 30$). In step 30, the trained VPGAN generated a frame F_{30} at T=30 conditioned on the 10 predicted frames F_{20}, F_{21}, ..., F_{29}. For the UCF101 dataset, we used 3 frames as context frames and predicted 8 frames as done in [38]. We draw and present performance charts where the X-axis represents different T values, and for each T value the Y-axis represents the SSIM (PSNR) value computed based on the ground truth and the frame predicted or generated in step T.

3.4.1 Evaluation of Cycle-Consistency Loss

We present in (3.9) the general formula of our multi (k steps) cycle-consistency loss function, L^k_{cycle}. Here we show experimentally how different k values affect the performance of VPGAN. We trained VPGAN (without the action control module) using different k values on the Moving Mnist dataset [89] with the objective loss as

defined in (3.12). The action control module was excluded because we were only interested in the impact of k on the behavior of VPGAN.

Fig. 3.7 shows the SSIM results and running time of VPGAN for different k values on the Moving Mnist dataset. (The PSNR results are similar and not shown here.) It can be seen from Fig. 3.7(a) that using L^k_{cycle} yields slightly more accurate results than using L^{k-1}_{cycle}. This is understandable given that L^{k-1}_{cycle} is part of L^k_{cycle}, as shown in (3.9). Enforcing earlier cycle consistency is undoubtedly more important than doing it k steps later (since k-steps cycle consistency couldn't be preserved if one-step cycle consistency isn't preserved). We decrease the value of a_i in (3.9) as i becomes larger. Therefore, as shown in Fig. 3.7(a), there isn't a big performance gap between L^{k-1}_{cycle} and L^k_{cycle} in terms of the SSIM measure. On the other hand, using L^k_{cycle} requires much more running time than using L^{k-1}_{cycle}, as shown in Fig. 3.7(b). The running time difference between L^{k-1}_{cycle} and L^k_{cycle} lies in the calculation of l^k_{cycle}, which requires $4k$ iterations of calculating G_θ. The total time complexity for L^k_{cycle} is $O(k^2)$ iterations of calculating G_θ. Similar results were obtained on the other datasets. After evaluating the trade-off between the model accuracy and running time, we decided to use L^1_{cycle} (i.e., $k = 1$) for our VPGAN framework in subsequent experiments since the accuracy difference between the different k values is not statistically significant ($p > 0.05$) according to the non-parametric Friedman test [39] while the time complexity grows with square.

3.4.2 Comparison of Image Segmentation Models

Here we compare the image segmentation models SegNet [5], U-Net [79] and DeepLabv3 [15] on different datasets. Fig. 3.8 presents the SSIM results on Moving Mnist and BAIR. (The PSNR and SSIM results on the other datasets are similar and not shown here.) In Fig. 3.8, VPGAN is our default model that is combined with SegNet. U-NetV is the model obtained by replacing SegNet in VPGAN with U-Net.

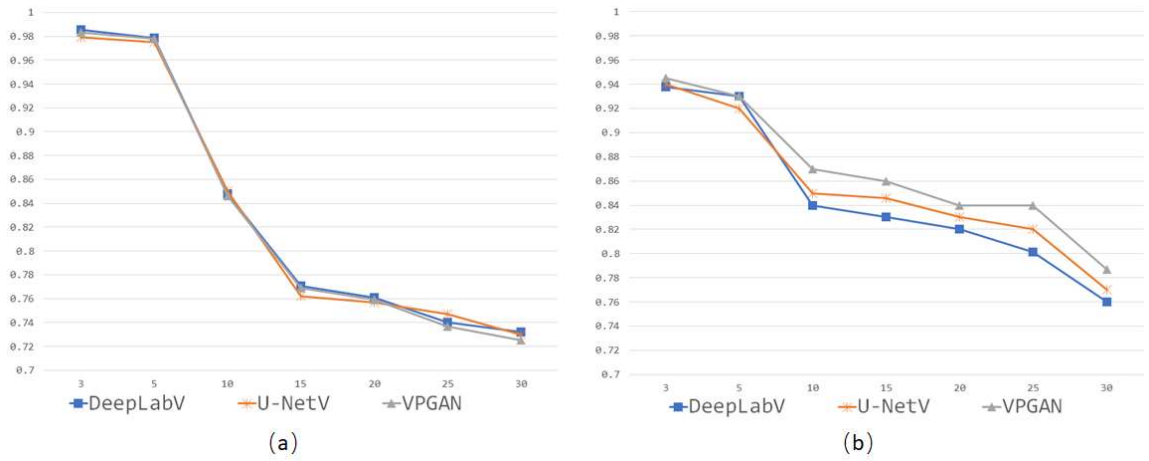

Figure 3.8 SSIM results of DeepLabV, U-NetV and VPGAN on two different datasets (a) Moving Mnist and (b) BAIR respectively. Given 10 context frames, the models predict 30 frames recursively, one by one. The X-axis represents different T (time step) values, and for each T value the Y-axis represents the SSIM value computed based on the ground truth and the frame predicted in step T.

DeepLabV is the model obtained by replacing SegNet in VPGAN with DeepLabv3. The accuracy difference between these models is not statistically significant ($p > 0.05$) on Moving Mnist while SegNet is the best on BAIR according to the Friedman test. As a result, we choose SegNet in our framework.

3.4.3 Ablation Studies

We performed ablation studies by considering four models: VPGAN, VPGAN-C representing VPGAN without our cycle-consistency loss function, VPGAN-S representing VPGAN without SegNet, and VPGAN-CS representing VPGAN without the cycle-consistency loss function and SegNet. Fig. 3.9 shows that the full model VPGAN achieves the best SSIM results on BAIR and KTH. (The PSNR and SSIM results on the other datasets are similar and not shown here.) According to the Friedman test, the difference between VPGAN and VPGAN-C (VPGAN-S, VPGAN-CS respectively) is statistically significant ($p < 0.05$). These results demonstrate the effectiveness of different components in VPGAN.

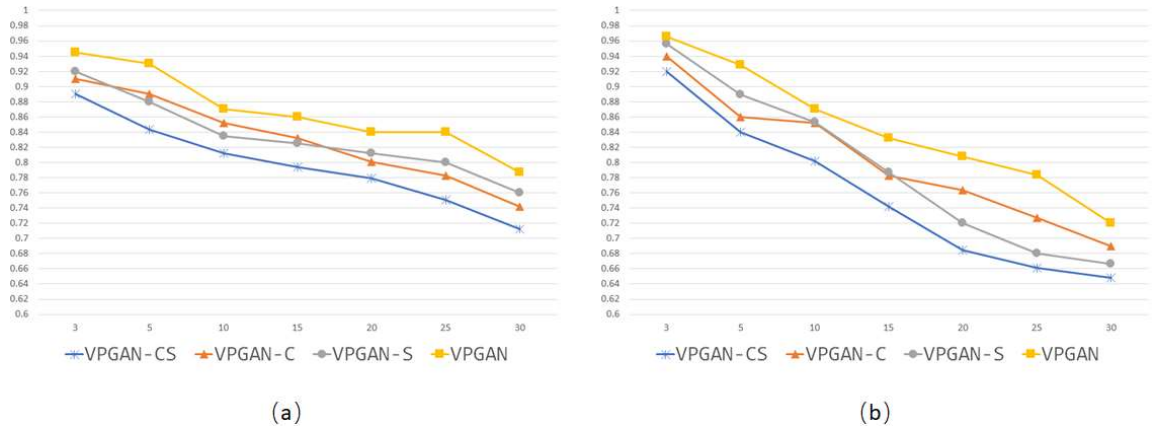

Figure 3.9 SSIM results of VPGAN-CS, VPGAN-C, VPGAN-S and VPGAN on two datasets (a) BAIR and (b) KTH respectively. Given 10 context frames, the models predict 30 frames recursively, one by one. The X-axis represents different T (time step) values, and for each T value the Y-axis represents the SSIM value computed based on the ground truth and the frame predicted in step T.

3.4.4 Results on the BAIR Dataset

The BAIR robot pushing dataset [27] involves a series of videos generated by a Sawyer robotic arm pushing a variety of objects. All of the videos have relatively similar surroundings (table settings) with a static background. Each video collects actions taken by the robotic arm corresponding to the commanded gripper pose. This dataset is very challenging for two reasons:

1. The movement is almost random and quite unpredictable.

2. It is a real-world video, with various objects and interactions between the robotic arm and objects (rather than a single frame-centered object with a neutral background).

The videos have a resolution of 64×64 pixels. Thus, our input dimension is $64 \times 64 \times 3$.

Fig. 3.10 presents some examples of frames generated by our approach on the BAIR dataset. The figure shows the potential of VPGAN in generating many possible future frames conditioned on different latent variables. It can be seen from the figure that all the random generations produced by our approach have the same good image

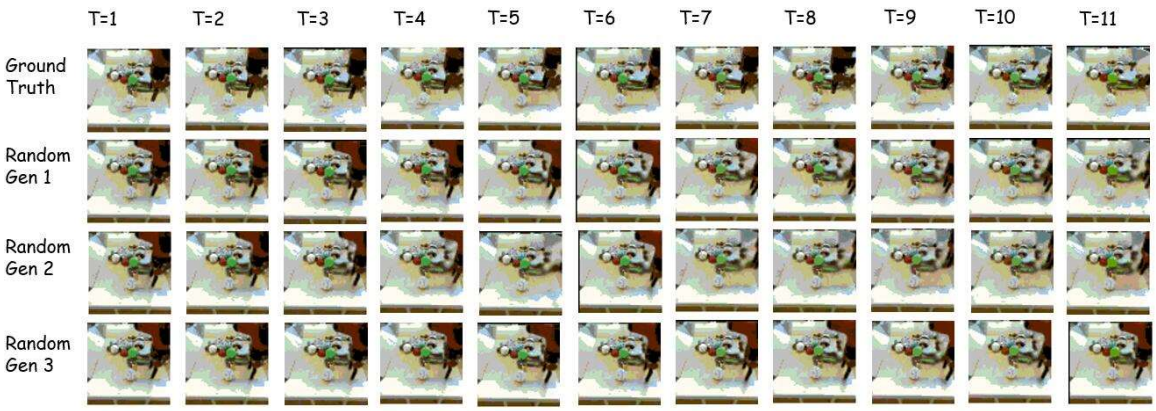

Figure 3.10 Examples of generated frames on the BAIR dataset to show the potential of VPGAN in generating many possible future frames.

quality as the ground truth. Fig. 3.11 compares our approach with SVG-LP [19] and SV2P [52] on the BAIR dataset. It is clear that the proposed VPGAN framework outperforms the related methods. The difference between VPGAN and the existing methods is statistically significant with for example $p = 0.0082$ between VPGAN and the current best SVG-LP based on SSIM and the Friedman test.

3.4.5 Results on the KTH Dataset

The KTH action dataset [81] contains various types of videos collected in real-world cameras. These videos include human subjects carrying out six activities (walking, jogging, running, boxing, hand waving, and hand clapping). For the first three activities, the human subject enters and leaves the frame multiple times, leaving the frame empty with a mostly static background for multiple frames at a time. Like the BAIR dataset, the videos in the KTH dataset have a resolution of 64×64 pixels.

Fig. 3.12 presents examples of frames generated by our approach on the KTH dataset. It can be seen from the figure that when T $\geq$ 10, the image quality of the particular 'human' drops significantly and the 'human' part becomes blurry. Nevertheless, the quality of the frames generated by our approach is as good as the ground truth for all the different T values in the figure. Fig. 3.13 compares

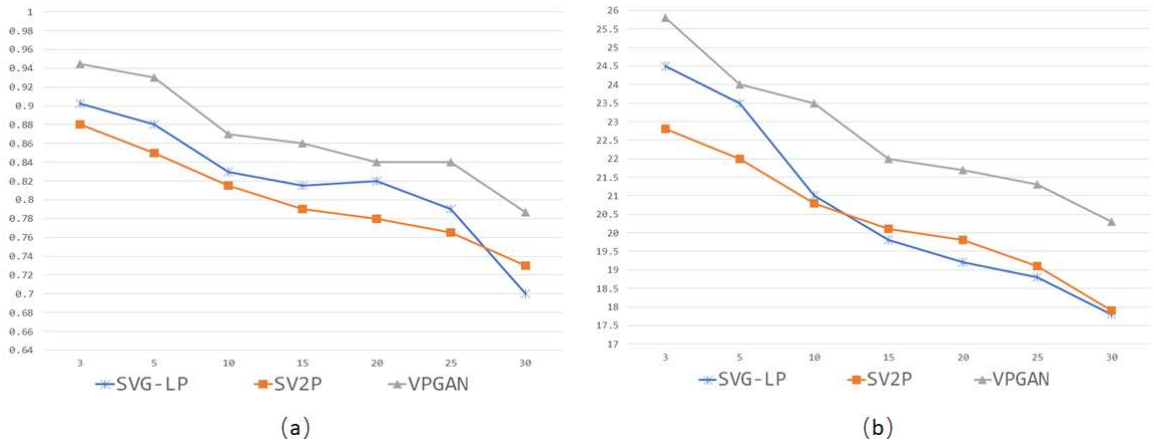

Figure 3.11 Comparison of SVG-LP, SV2P and VPGAN on the BAIR dataset based on (a) SSIM and (b) PSNR measures. Given 10 context frames, the models predict 30 frames recursively, one by one. The X-axis represents different T (time step) values, and for each T value the Y-axis represents the (a) SSIM value and (b) PSNR value computed based on the ground truth and the frame predicted in step T.

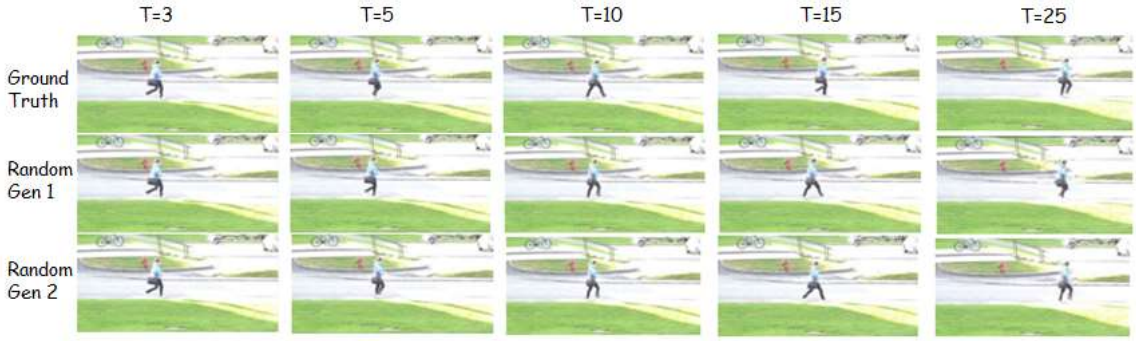

Figure 3.12 Examples of frames generated by VPGAN on the KTH dataset.

our approach with SVG-LP [19], SV2P [52] and BCnet-D [38] on the KTH dataset. Again, it is evident that the proposed VPGAN framework performs better than the existing methods. The difference between VPGAN and the existing methods is statistically significant with for example $p = 0.0253$ between VPGAN and the current best BCnet-D based on SSIM and the Friedman test.

3.4.6 Results on the UCF101 Dataset

UCF101 [88] contains realistic action videos collected from YouTube. It has 13320 videos from 101 action categories. It is a very challenging dataset and gives the

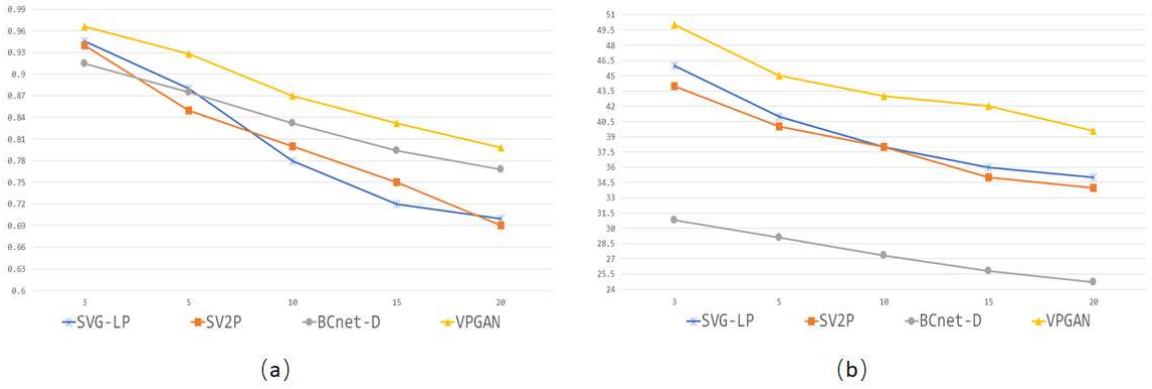

Figure 3.13 Comparison of SVG-LP, SV2P, BCnet-D and VPGAN on the KTH dataset based on (a) SSIM and (b) PSNR measures. Given 10 context frames, the models predict 20 frames recursively, one by one. The X-axis represents different T (time step) values, and for each T value the Y-axis represents the (a) SSIM value and (b) PSNR value computed based on the ground truth and the frame predicted in step T.

largest diversity in terms of actions and with the presence of large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, and so on [88]. In evaluating our method on the UCF101 dataset, we conditioned VPGAN on 3 frames to generate 8 frames, resized with a resolution of 240×320 pixels, as done in [38].

Fig. 3.14 presents examples of frames generated by our approach on the UCF101 dataset. It can be seen from the figure that our approach maintains accurate results in the first 5 predictions though the human part becomes fuzzy in the latter 7-8 predictions. Fig. 3.15 compares our approach with BCnet-D [38] and MCnet+Res [94] on the UCF101 dataset. It can be seen from the figure that the proposed VPGAN framework performs better than BCnet-D and MCnet+Res. The difference between VPGAN and the existing methods is statistically significant with for example $p = 0.0047$ between VPGAN and the current best BCnet-D based on SSIM and the Friedman test.

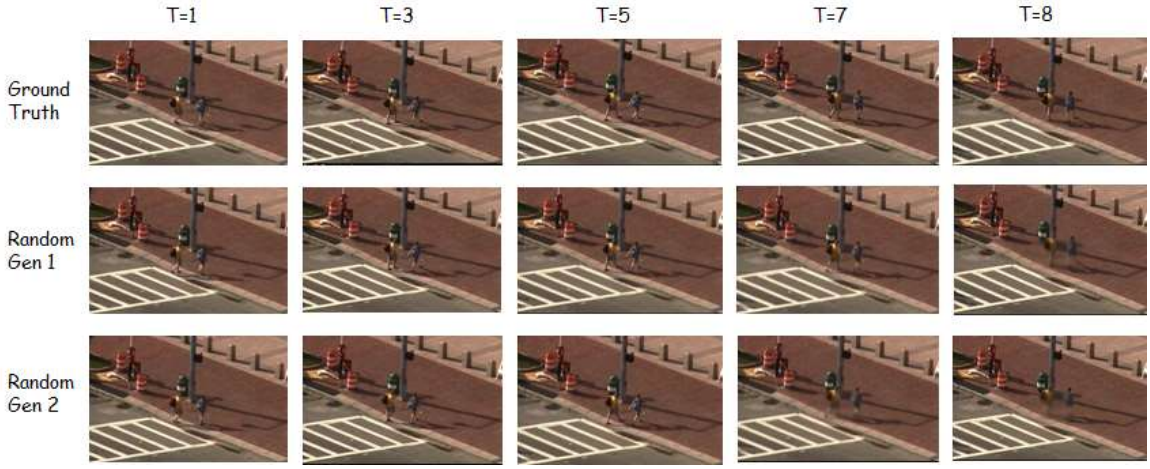

Figure 3.14 Examples of frames generated by VPGAN on the UCF101 dataset.

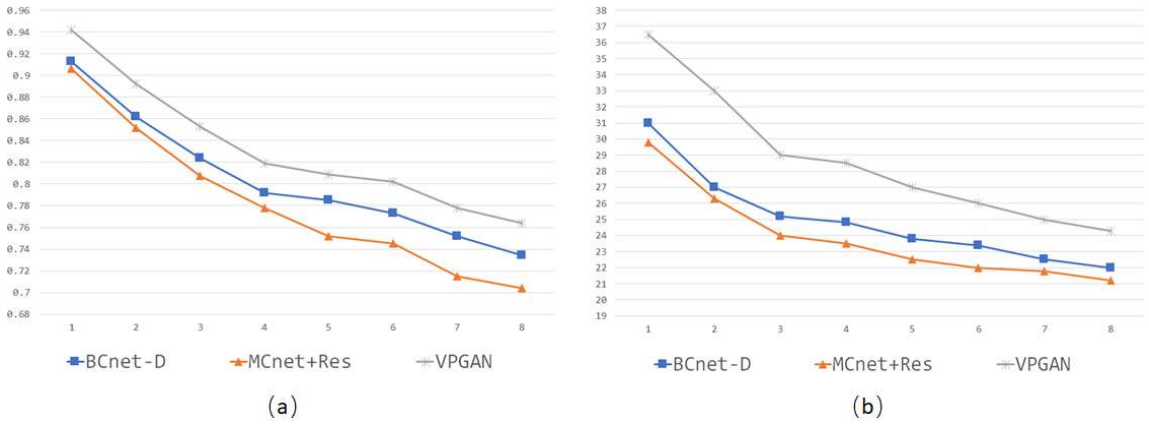

Figure 3.15 Comparison of BCnet-D, MCnet+Res and VPGAN on the UCF101 dataset based on (a) SSIM and (b) PSNR measures. Given three context frames, the models predict eight frames recursively, one by one. The X-axis represents different T (time step) values, and for each T value the Y-axis represents the (a) SSIM value and (b) PSNR value computed based on the ground truth and the frame predicted in step T.

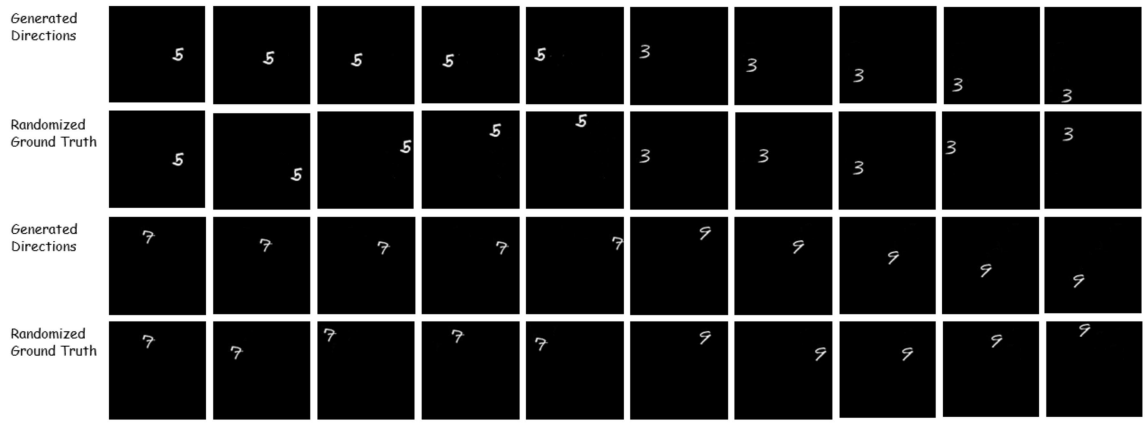

Figure 3.16 Generating desired movements of Mnist characters. '5' is moving toward the left. '3' is moving downward. '7' is moving toward the right. '9' is moving right-down.

3.4.7 Evaluation of Action Control

In Section 3.3.5, we introduce the network τ to accomplish action control. Since τ is a conformal mapping, in theory, it preserves angles between the latent variable space and the moving direction space. However, in practice, its accuracy is affected by the training procedure. We used the Moving Mnist dataset [89] to evaluate τ. The reason for choosing this simple dataset is that we were mainly interested in the potential of action control with our techniques. Moving Mnist could well reflect the movement of a single character and doesn't involve complicated environmental changes.

Fig. 3.16 presents some frames generated by our VPGAN on Moving Mnist. Our 'next frame' was generated by choosing a specific direction and executing Algorithm 1. By comparing the generated frames and ground truth in Fig. 3.16, we can see that each character in Moving Mnist is actually moving around the space randomly, but by executing Algorithm 1, we can gain action control of the character.

3.5 Summary

In this research, we present an adversarial inference framework (VPGAN) for stochastic video prediction, and incorporate cycle consistency and conformal mapping

into our VPGAN framework. Cycle consistency relieves the problem of blurry predictions to obtain more accurate results while conformal mapping enables action control through manipulating latent variables. Our experimental results show that the proposed VPGAN approach works well on different datasets and outperforms existing methods when combined with SegNet.

In future works we plan to extend the VPGAN framework for video processing in scientific domains (e.g., solar physics). In solar physics, deep learning has drawn a lot of interest due to its effectiveness in processing big and complex observational data gathered from diverse instruments [57]. Video is the most common form of observational data. We plan to use VPGAN to predict solar videos and compare them with the observational data, and also use VPGAN to enhance the quality of the observational data.

CHAPTER 4

DEEP LEARNING-BASED SYNTHESIS OF VECTOR MAGNETOGRAMS USING GENERATIVE ADVERSARIAL NETWORKS

4.1 Background

Deep learning is a branch of machine learning where neural networks are designed to learn from large amounts of data [50]. It has been used extensively in computer vision, natural language processing, and lately in biology [25, 100], medicine [7, 104], heliophysics [30, 58, 59], astronomy [45, 60], and so on. Deep learning employs various networks such as deep neural networks, deep belief networks, convolutional neural networks and recurrent neural networks, among which generative adversarial networks (GANs) have draw significant interest in recent years [33]. In a Generative Adversarial Network (GAN) model, two neural networks, called the generator and discriminator respectively, contest with each other in a zero-sum game. The generator generates fake samples while the discriminator evaluates the fake samples. The contest operates in terms of data distributions. The generator learns to map from a latent space to a data distribution while the discriminator distinguishes the fake samples produced by the generator from the true data distribution. The generator's objective is to fool the discriminator by producing fake samples that the discriminator thinks are part of the true data distribution. When this objective is accomplished, the training of the GAN model is completed. GANs have been used in video prediction [41], image enhancement [18], image-to-image translation [42], and image generation (synthesis) [45, 60].

For example, Kim et al. generated farside solar magnetograms from STEREO Extreme UltraViolet Imager (EUVI) 304-Å images using a deep learning model based

on conditional generative adversarial networks (cGANs) [45]. The authors trained their model using pairs of Solar Dynamics Observatory (SDO)/Atmospheric Imaging Assembly (AIA) 304-Å images and SDO/Helioseismic and Magnetic Imager (HMI) magnetograms. They reported some preliminary results obtained from the cGAN model [45]. Liu et al. performed a more detailed analysis of the cGAN model and concluded that more research needs to be performed to obtain scientifically reliable magnetograms [60].

Inspired by Kim et al.'s work, we propose here a new deep learning model, dubbed MagGAN, to generate vector magnetograms. The cGAN model, whose architecture was developed for image-to-image translation in computer vision, is not applicable to our work [42]. Instead, we design a novel architecture and loss function tailored for vector magnetograms generation to take into account the magnetic field strength. We apply MagGAN to generate synthetic magnetic field components B'_x and B'_y. The synthetic B'_x and B'_y along with the line-of-sight (LOS) component of the magnetic field, which can be treated as B_z, create vector magnetograms.

Our work is motivated by the observation that solar cycle 24 (from 2008 to 2019) has been relatively weak, though it is the only solar cycle in which consistent vector magnetograms are available through SDO/HMI where SDO was launched in 2010. We therefore intend to study solar cycle 23 (from 1996 to 2008), which had stronger solar flare activity. Specifically, we consider a major data archive, namely the Solar and Heliospheric Observatory (SOHO)/Michelson Doppler Imager (MDI) archive with data from 1996 to 2011, which covers the more active solar cycle 23 [80]. However, SOHO/MDI data only has line-of-sight (LOS) magnetograms, without vector magnetograms. By using SOHO/MDI LOS magnetograms and Hα observations (images) obtained from the Big Bear Solar Observatory (BBSO), our MagGAN is able to generate synthetic magnetic field components B'_x and B'_y, and hence create synthetic vector magnetograms for solar cycle 23. These generated

vector magnetograms will be useful for analyzing and forecasting solar flare activity. For example, one can derive magnetic field parameters from the vector magnetograms and use the magnetic field parameters to predict the occurrence of solar flares within next 24 hours [56, 58]. It should be pointed out that since we are mainly interested in solar flares, we cut active regions (ARs) patches of 256×256 pixels that may produce flares from the LOS magnetograms. Here, an AR is a connected component in which the magnetic field strength of some pixel must be greater than 800 Gauss. We then align the cut ARs patches with the corresponding regions in the Hα images. Finally, we use the aligned pairs of regions to train MagGAN to generate synthetic B'_x, B'_y in the ARs. Thus, our approach differs from the work of Kim et al., who performed the alignment of pairs of full-disk images.

4.2 Methods

4.2.1 Datasets

As mentioned above, the LOS magnetograms obtained by SOHO/MDI can date back to the year of 1996. Thus, SOHO/MDI and its successor, SDO/HMI, provide LOS magnetograms with the coverage of the most recent two solar cycles (#23 and #24) starting from 1996. In addition, SDO/HMI has provided vector magnetograms of the recent ten years starting from May 1, 2010. Specifically, HMI observes the full solar disk at 6173 Å with a resolution of 1" (arcsecond) and cadence of 12 minutes. Meanwhile, BBSO has provided Hα observations (images) since 1970s. BBSO's full-disk Hα observations are taken every $\sim$30 minutes, up to 6 hours, for one observing day at the wavelength of 6563 Å with the same resolution. Unlike satellite-based instruments such as MDI and HMI, BBSO is a ground-based telescope that sometimes has seeing limitations due to unstable conditions of Earth's atmosphere and weather. In this study, we excluded low-quality Hα images with an incomplete field of view

(FOV) and cloud shades as well as other out-of-focus images. Then, we selected 3757 high-quality BBSO Hα images in the period from 2011-04-12 to 2017-12-31.

Our training set comprised the selected 3757 Hα images from 2011-04-12 to 2017-12-31 and their temporarily closest HMI LOS magnetograms and HMI vector magnetograms. For each selected Hα image, there were both HMI LOS magnetograms and HMI vector magnetograms available within 6 minutes of the Hα image. We cut active regions (ARs) that may produce flares from the HMI LOS magnetograms and B_x, B_y of the HMI vector magnetograms in the training set. Totally, there were 8691 cut ARs from the LOS magnetograms, B_x and B_y respectively. Furthermore, we aligned the cut ARs with the corresponding regions in the temporarily closest training Hα images and 85% of these aligned regions were used to train MagGAN.

MDI and HMI have an overlapping period from 2010-05-01 to 2011-04-11. Our test set thus comprised the above 15% remaining ARs and selected 100 Hα images from 2010-05-01 to 2011-04-1 and their temporarily closest MDI LOS magnetograms in the overlapping period. Here, we cut active regions (ARs) that may produce flares from the MDI LOS magnetograms in the test set and aligned the cut ARs with the corresponding regions in the temporarily closest testing Hα images. There were totally 350 pairs of aligned regions. We input these pairs of aligned regions to the trained MagGAN model to generate synthetic B'_x and B'_y, each with 256×256 pixels. The ground truth data of B_x and B_y obtained from HMI vector magnetograms were used to validate whether our predicted B'_x and B'_y were good or not.

4.2.2 Overview of MagGAN

Generating synthetic B'_x and B'_y amounts to solving a regression problem because the output of the generating procedure consists of real numbers, i.e., magnetic field strengths. Since the active regions of HMI LOS magnetograms (images) contain complex and diverse patterns, we propose a new GAN architecture composed of a

novel convolutional neural network (CNN) [51] with self-attention [103] to solve this regression problem. Figure 4.1 presents an overview of the entire MagGAN-learning (training) process specifically for generating synthetic B'_x where the discriminator, denoted D_x, and generator, denoted G_x, competes and learns in an adversarial way for training the following loss function, denoted L^x_{adv}, to reach an equilibrium:

$$\begin{aligned} \min_{G_x}\max_{D_x} L^x_{adv}(G_x, D_x) &= E_{B_x\sim P_{DATA}(B_x)}[\log D_x(\text{LOS}, B_x)] \\ &+ E_{B'_x\sim P_{DATA}(B_x)}[\log(1 - D_x(\text{LOS}, B'_x))] \\ B'_x &= G_x(\text{LOS}, \text{H}\alpha) \end{aligned} \tag{4.1}$$

Here, E takes the expectation over $P_{DATA}(B_x)$, which is the true distribution of B_x. The generator G_x takes as input an aligned pair of HMI LOS and Hα images and generates as output the magnetic field component B'_x. The discriminator D_x takes as input the pair of HMI LOS and B'_x (B_x, respectively) images and produces as output "Fake" features ("Truth" features, respectively) for the fake sample B'_x (the ground truth B_x, respectively). Initially, $D_x(\text{LOS}, B_x) = 1$ and $D_x(\text{LOS}, B'_x) = 0$. When the equilibrium is reached, the training is completed where the discriminator cannot tell the difference between the generated fake sample and ground truth. We adopt the same training scheme for generating the magnetic field component B'_y with the generator G_y, discriminator D_y, and loss L^y_{adv}.

Figure 4.2(a) illustrates the generator Gx; G_y is implemented similarly and its description is omitted. The aligned pair of HMI LOS and Hα images is first fed to a ResNet-like structure [37]. The output of the ResNet is then distributed through fully connected layers to two modules: the channel attention module (CAM) and position attention module (PAM), both of which were originally designed for DANet [28]. CAM and PAM leverage the self-attention mechanism to better capture and

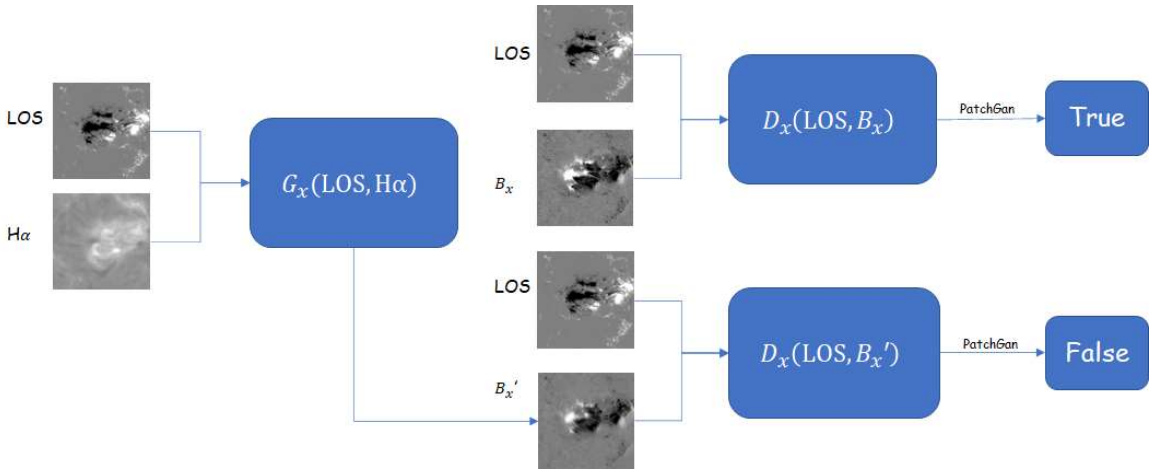

Figure 4.1 Overview of the training process of MagGAN. LOS represents an active region of an HMI line-of-sight magnetogram, Hα represents the corresponding region of the Hα image temporarily closest to the LOS, B_x is the ground truth and B'_x is the generated fake sample.

transform a wider range of contextual information into local features, thus enhancing their representation capability. Both CAM and PAM are calculated in a similar way where CAM applies the self-attention mechanism to image channels and PAM focuses on the calculation of location information. The outputs of CAM and PAM are then combined and sent to two convolutional blocks to produce the magnetic field component B'_x.

The discriminator D_x employs the same architecture as the generator G_x except that D_x takes as input the pair of HMI LOS and B'_x (B_x, respectively) images and produces as output "Fake" features ("Truth" features, respectively) for the fake sample B'_x (the ground truth B_x, respectively); see Figure 4.2(b). The discriminator D_y is implemented similarly and its description is omitted.

During inference (testing), MagGAN takes as input an aligned pair of MDI LOS and Hα images and predicts as output a fake sample B'_x (B'_y, respectively) through the trained generator G_x (G_y, respectively). These B'_x and B'_y are generated magnetic field components corresponding to the input MDI LOS and Hα images.

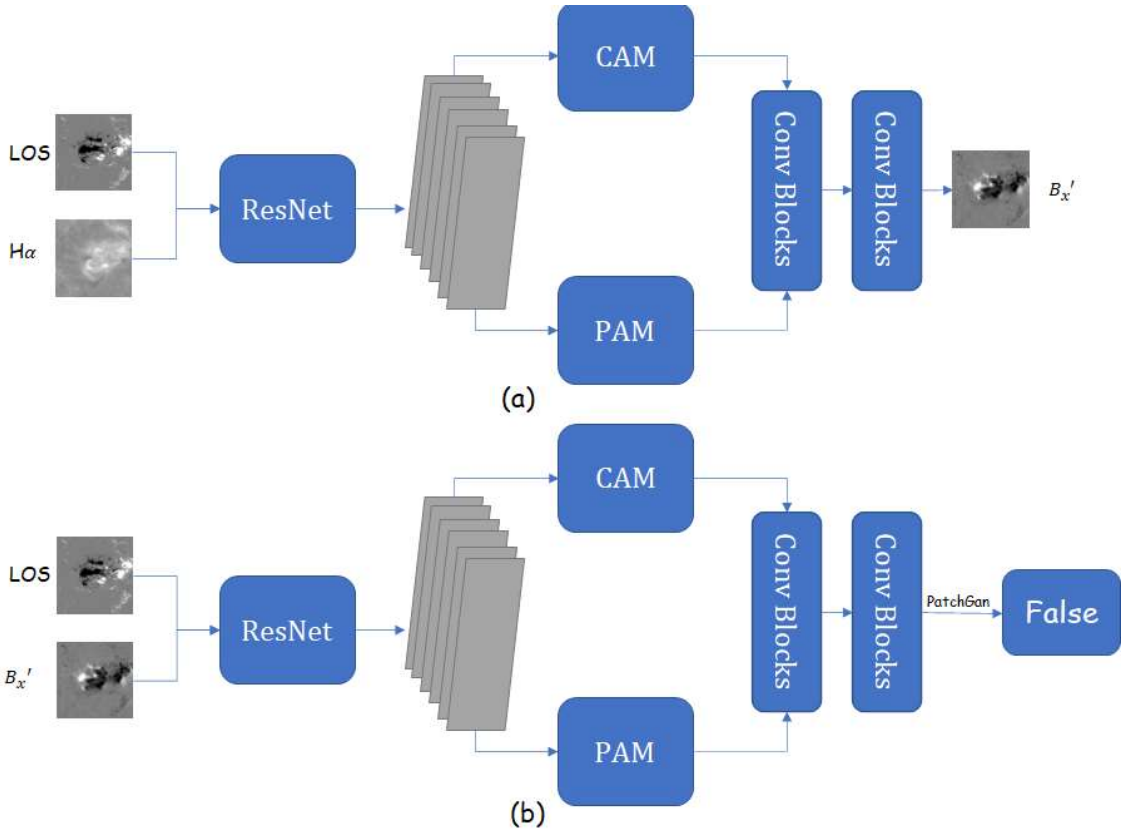

Figure 4.2 (a) Illustration of the generator G_x of MagGAN, which takes as input the aligned pair of HMI LOS and Hα images and generates as output the fake sample B'_x. (b) Illustration of the discriminator D_x of MagGAN, which takes as input the pair of HMI LOS and B'_x images and produces as output "Fake" features for the fake sample B'_x.

4.2.3 Loss Function

A large portion of an active region (AR) has low magnetic field strengths where the magnetic field strength of a pixel is smaller than 200 Gauss. Relatively few pixels in an AR have magnetic field strengths larger than 200 Gauss. To tackle this imbalanced problem, we propose a novel weighted L_r loss of a pixel p, defined as:

$$L_r^p(s', s) = |\frac{s}{c}||s' - s| \tag{4.2}$$

Here, s' represents the generated magnetic field strength at p, s represents the true magnetic field strength at p, and c is a threshold. The absolute difference between s' and s at p, usually reflected by the L_1 loss, is multiplied by a weight, $|\frac{s}{c}|$. This suggests that a pixel p with a larger (smaller, respectively) magnetic field strength yield a larger (smaller, respectively) L_r^p loss.

Let μ_x (σ_x, respectively) denote the mean (variance, respectively) of the magnetic field strengths of all pixels in B_x. Let μ_y (σ_y, respectively) denote the mean (variance, respectively) of the magnetic field strengths of all pixels in B_y. Let

$$c_x = \frac{200 - \mu_x}{\sigma_x}, \quad c_y = \frac{200 - \mu_y}{\sigma_y} \tag{4.3}$$

The weighted L_r loss between the generated fake sample B'_x (B'_y, respectively) and the ground truth B_x (B_y, respectively) is denoted by $L_r(B'_x, B_x)$ ($L_r(B'_y, B_y)$, respectively), in which the L_r^p loss of each pixel p in B'_x and B_x (B'_y and B_y, respectively) is defined as in Equation (4.2) and the threshold c equals c_x (c_y, respectively). The total loss, denoted L_{mag}, is defined as the sum of the adversarial loss in Equation (4.1) and the weighted L_r loss, as shown in Equation (4.4) below:

$$L_{mag} = 0.2 \times (L_{adv}^x + L_{adv}^y) + L_r(B'_x, B_x) + L_r(B'_y, B_y) \tag{4.4}$$

The training of MagGAN is conducted by applying the Adam [6, 108] optimizer to minimizing the total loss, L_{mag}, until an equilibrium is reached. In our experiments, we use a batch size of 2 to save memory and train MagGAN with 35 epochs in generating the magnetic field components on an NVIDIA GeForce RTX 2080 GPU machine.

4.3 Results

4.3.1 Performance Metrics

We conducted a series of experiments to evaluate the performance of MagGAN based on four performance metrics: mean absolute error (MAE) [82], percent agreement (PA) [64], R-squared [82] and Pearson product-moment correlation coefficient (PPMCC) [29, 72]. For each magnetic field component, we compared its ground truth values with our MagGAN-predicted values and computed the four performance metrics.

The first performance metric is defined as:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|s_i - s_i'|, \tag{4.5}$$

where $N = 256 \times 256 = 65536$ is the total number of pixels in a magnetic field component, and s_i (s_i', respectively) denotes the true (MagGAN-predicted, respectively) magnetic field strength for the ith pixel, $1 \leq i \leq 65536$, in the magnetic field component. This metric is used to quantitatively assess the dissimilarity (distance) between the ground truth values and MagGAN-predicted values in the magnetic field component.

The second performance metric is defined as:

$$\text{PA} = \frac{M}{N} \times 100\%, \tag{4.6}$$

where M denotes the total number of agreement pixels in a magnetic field component. We say the ith pixel in the magnetic field component is an agreement pixel if $|s_i - s_i'|$ is smaller than a user-specified threshold. (The default threshold is set to 200 Gauss for B_{total}, B_x, B_y, B_z respectively.) This metric is used to quantitatively assess the

similarity between the ground truth values and MagGAN-predicted values in the magnetic field component.

The third performance metric is defined as:

$$\text{R-squared} = 1 - \frac{\sum_{i=1}^{N}(s_i - s_i')^2}{\sum_{i=1}^{N}(s_i - \overline{s})^2}, \tag{4.7}$$

where $\overline{s} = \frac{1}{N}\sum_{i=1}^{N} s_i$ denotes the mean of the ground truth values for all the pixels in a magnetic field component. The R-squared value, ranging from $-\infty$ to 1, is used to measure the strength of the relationship between the ground truth values and MagGAN-predicted values in the magnetic field component. The larger (i.e., the closer to 1) the R-squared value is, the stronger relationship between the ground truth values and MagGAN-predicted values we have.

The fourth performance metric is defined as:

$$\text{PPMCC} = \frac{\text{E}[(T - \mu_T)(G - \mu_G)]}{\sigma_T \sigma_G}, \tag{4.8}$$

where T and G represent the ground truth values and MagGAN-predicted values, respectively, in a magnetic field component, μ_T and μ_G are the mean of T and G respectively, σ_T and σ_G are the standard deviation of T and G respectively, and $E(\cdot)$ is the expectation. The value of PPMCC ranges from -1 to 1. A value of 1 means that a linear equation describes the relationship between T and G perfectly where all data points lying on a line for which G increases as T increases. A value of -1 means that all data points lie on a line for which G decreases as T increases. A value of 0 means that there is no linear correlation between the variables T and G. PPMCC measures the linear correlation between the ground truth values and

MagGAN-predicted values, quantifying how close these values are [29, 72, 82]. Notice that PA, R-squared and PPMCC do not have units while MAE has a unit (Gauss).

4.3.2 Experimental Results

In this section, we evaluate the performance of MagGAN and present our experimental results. Based on the test set described in the "Datasets" subsection, our MagGAN consists of two different sets of data, 214 images of the Helioseismic Magnetic Imager (HMI) and about 100 images of the Michelson Doppler Imager dataset (MDI).

There are totalyl 350 pairs of true magnetic field components B_x and B_y, each also with 256×256 pixels. Table 4.1 presents the average performance metric values of MagGAN based on the data in the test set. We separate the results of MDI and HMI as format MDI :, HMI :. The table shows that the overall results of synthetic B'_x and B'_y perform very close (< 100) to the true B_x and B_y for both MDI and HMI. Notice that there is a slight degradation in the results of MDI compared to HMI, mainly due to the lower resolution of the MDI instrument and larger time gaps between MDI images and corresponding Hα images.

Magnetic Field Components	MAE	PA	R-squared	PPMCC
MDI: B_x, B'_x	85.98	85.13%	0.6254	0.8124
MDI: B_y, B'_y	87.56	82.17%	0.6678	0.7956
HMI: B_y, B'_y	63.23	90.23%	0.7824	0.8623
HMI: B_y, B'_y	66.34	91.45%	0.7921	0.8312

Table 4.1 Average Performance Metric Values of the MagGAN Model Based on the Data in Our Test Set

In addition to quantitative results, we next show some visualization results of MDI. Figure 4.3 presents results for a simple active region. It compares MagGAN-predicted magnetic field components B'_x, B'_y and true magnetic field components B_x, B_y where the magnetic field components, selected from the test set, contain relatively simple patterns. The first column shows MagGAN-predicted B'_x (top) and B'_y (bottom), each with 256×256 pixels. The second column shows true B_x (top) and

B_y (bottom), each also with 256×256 pixels. The third column shows scatter plots. The X-axis and Y-axis in each scatter plot represent true magnetic field strengths and MagGAN-predicted magnetic field strengths respectively. The diagonal line in each scatter plot corresponds to pixels whose true magnetic field strengths are identical to MagGAN-predicted magnetic field strengths. It can be seen from Figure 4.3 that the MagGAN-predicted magnetic field strengths are very close to the true magnetic field strengths.

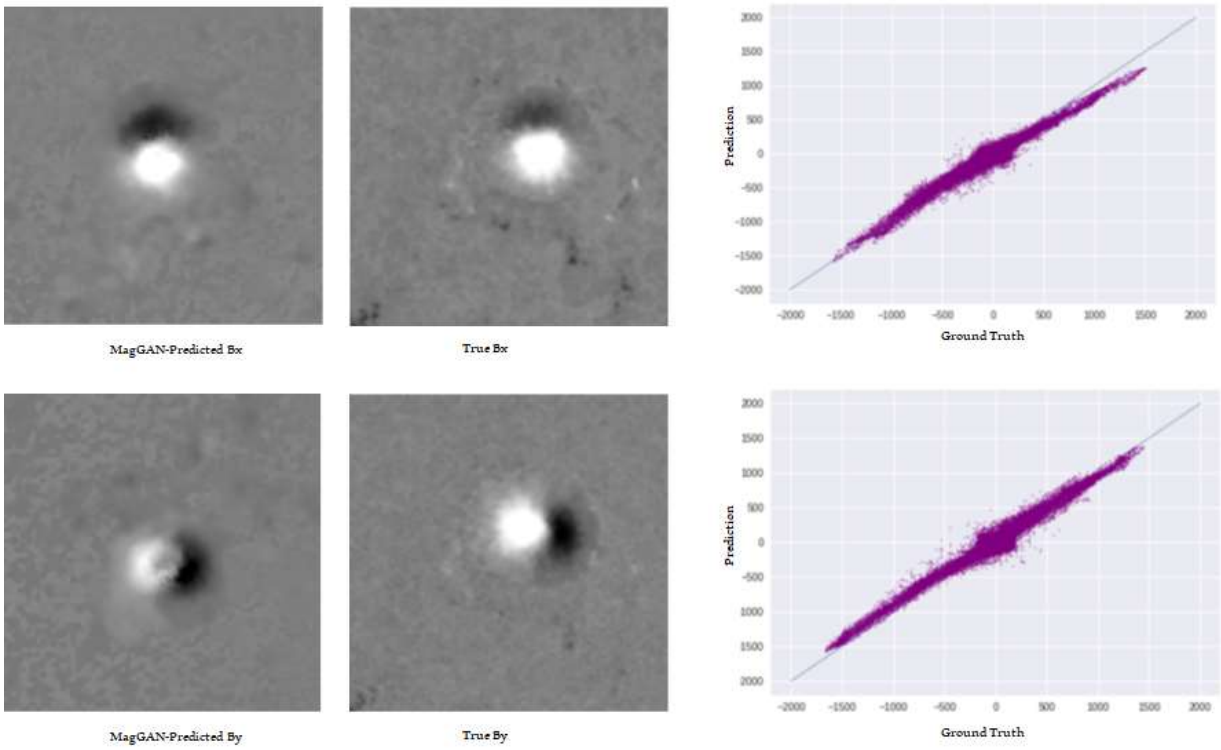

Figure 4.3 Comparison between MagGAN-predicted magnetic field components B'_x, B'_y and true magnetic field components B_x, B_y where the magnetic field components, selected from the test set, contain relatively simple patterns. The first column shows MagGAN-predicted B'_x (top) and B'_y (bottom). The second column shows true B_x (top) and B_y (bottom). The third column shows scatter plots. The X-axis and Y-axis in each scatter plot represent true magnetic field strengths and MagGAN-predicted magnetic field strengths respectively. The diagonal line in each scatter plot corresponds to pixels whose true magnetic field strengths are identical to MagGAN-predicted magnetic field strengths.

Figure 4.4 compares MagGAN-predicted magnetic field components B'_x, B'_y and true magnetic field components B_x, B_y where the magnetic field components, selected

from the test set, contain relatively complex patterns. It can be seen from Figure 4.4 that the MagGAN-predicted magnetic field strengths and the true magnetic field strengths are also highly correlated, though they are not as close to as those in Figure 4.3.

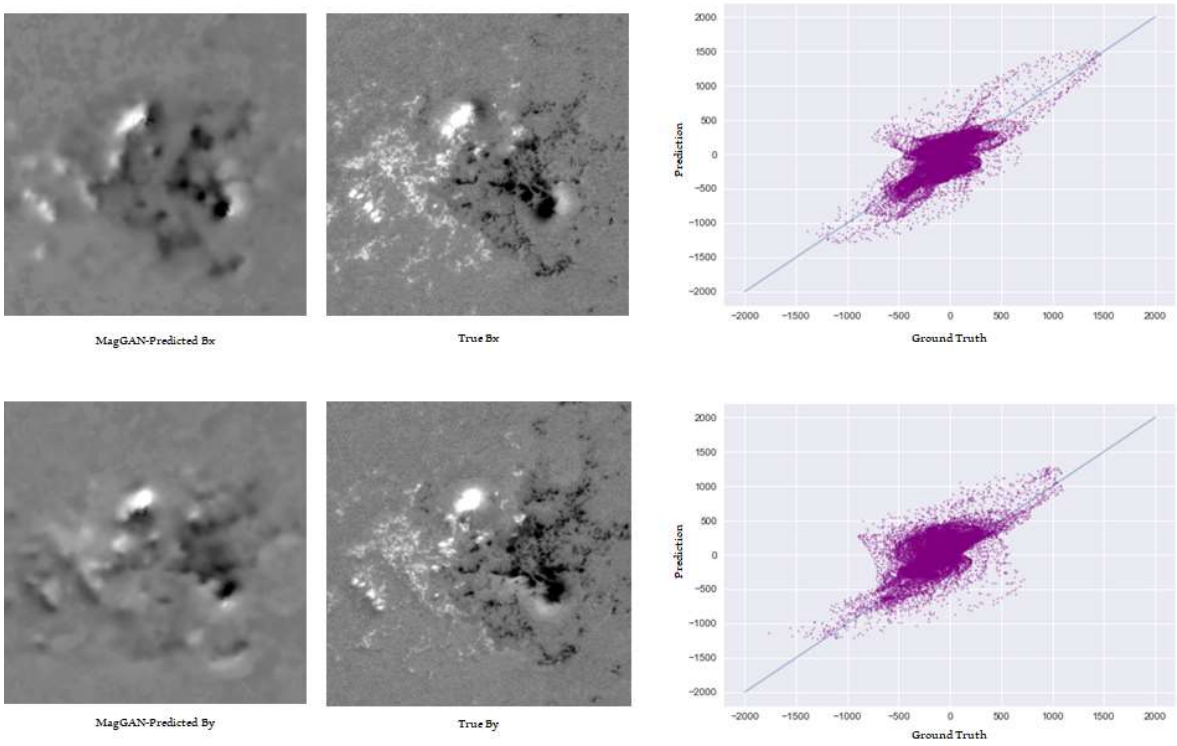

Figure 4.4 Comparison between MagGAN-predicted magnetic field components B'_x, B'_y and true magnetic field components B_x, B_y where the magnetic field components, selected from the test set, contain relatively complex patterns. The first column shows MagGAN-predicted B'_x (top) and B'_y (bottom). The second column shows true B_x (top) and B_y (bottom). The third column shows scatter plots. The X-axis and Y-axis in each scatter plot represent true magnetic field strengths and MagGAN-predicted magnetic field strengths respectively. The diagonal line in each scatter plot corresponds to pixels whose true magnetic field strengths are identical to MagGAN-predicted magnetic field strengths.

We have shown above sample results for MDI dataset. Below we present prediction results on the HMI dataset. We selected a complex region in year 2017, and applied MagGan to this region.

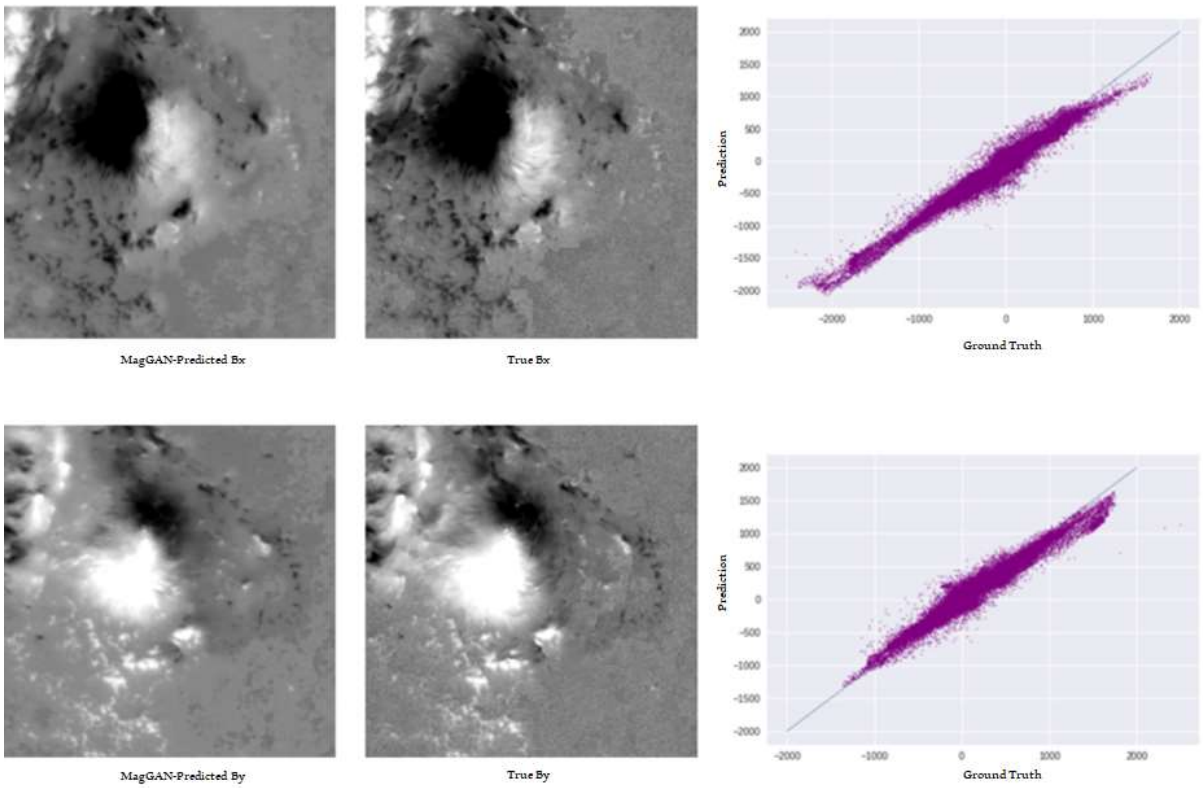

Figure 4.5 Comparison between MagGAN-predicted magnetic field components B'_x, B'_y and true magnetic field components B_x, B_y where the magnetic field components, selected from the test set, contain relatively complex patterns. The first column shows MagGAN-predicted B'_x (top) and B'_y (bottom). The second column shows true B_x (top) and B_y (bottom). The third column shows scatter plots. The X-axis and Y-axis in each scatter plot represent true magnetic field strengths and MagGAN-predicted magnetic field strengths respectively. The diagonal line in each scatter plot corresponds to pixels whose true magnetic field strengths are identical to MagGAN-predicted magnetic field strengths.

Figure 4.5 compares relatively complex patterns between ground truth and our predicted results. Quantitative results in Table 4.1 indicate that the predictions on the HMI test dataset are more accurate than the predictions on the MDI test dataset, which is well reflected in Figure 4.5. The predictions and scatter plots in Figure 4.5 look more similar and accurate compared to the results in Figure 4.4 where both figures present results for complex regions. This happens because of the difference between the HMI instrument and the MDI instrument. MDI data have

lower resolution and larger time gaps. Thus, the model trained on HMI data would achieve higher error rates when tested on MDI data.

So far we have considered B_x and B_y separately. Next, we evaluate our model's performance by combining B_x and B_y together. Based on the test set, MagGAN generated 350 pairs of synthetic B'_x and B'_y. Figure 4.6 shows the average prediction errors of MagGAN as a function of magnetic field strengths based on the 350 pairs of synthetic B'_x and B'_y. For each magnetic field strength, k, the corresponding prediction error δ is defined as:

$$\delta = |\sqrt{s_{p,x}^2 + s_{p,y}^2} - \sqrt{(s'_{p,x})^2 + (s'_{p,y})^2}| \tag{4.9}$$

where $s_{p,x}$ ($s_{p,y}$, respectively) represents the true magnetic field strength for a pixel p in B_x (B_y, respectively) such that $\sqrt{s_{p,x}^2 + s_{p,y}^2}$ equals k, and $s'_{p,x}$ ($s'_{p,y}$, respectively) represents the MagGAN-predicted magnetic field strength for the same pixel p in B'_x (B'_y, respectively). It can be seen from Figure 4.6 that the prediction errors are bounded and do not exceed 190 Gauss even when magnetic field strengths are larger than 1600 Gauss. This happens because the loss function used by MagGAN takes into consideration the magnetic field strengths of pixels in active regions (ARs). The results in Figure 4.6 are encouraging since significant flares are likely to occur in ARs with very large magnetic field strengths. Our MagGAN tool can make pretty accurate predictions in these ARs, and hence will help to produce reliable flare forecasting results.

4.4 Summary

We propose a new deep learning approach (MagGAN) for generating synthetic magnetic field components to create synthetic vector magnetograms. This approach allows us to extend the active region (AR) coverage with vector magnetograms for

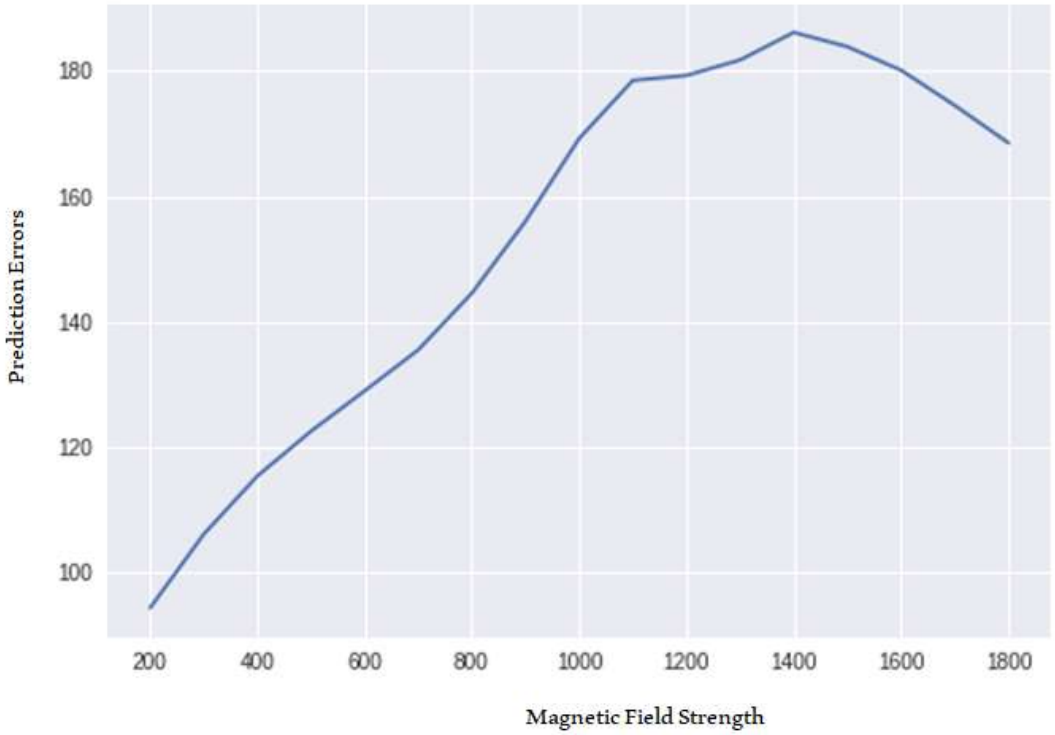

Figure 4.6 Average prediction errors of MagGAN as a function of magnetic field strengths based on the data in our test set.

both of solar cycles 23 and 24. One can then derive useful magnetic field parameters from the vector magnetograms to forecast solar flare activity [56, 58, 59]. Our experimental results show that the synthetic magnetic field components generated by MagGAN are very close to the ground truth in our test dataset. This good performance is due to MagGAN's innovative architecture and loss function tailored for vector magnetogram generation. Thus, we conclude that the proposed approach is feasible, and opens new directions for using AI-generated data to perform heliophysics research.

CHAPTER 5

CONCLUSION

The work and research of this dissertation aims mainly to solve three real-world application problems. Our novel architecture design not only enable us to outperform many existing methodology but also may introduce new features into these applications.

In the work of background subtraction, we're the first to introduce three dimensional convolution and stacked-ConvLSTM into background subtraction problem. We created a big gap in accuracy compared to previous simple deep learning model.

The following work of video prediction, our proposed methodology of embedding conformal mapping into the feedforward inference enables the new feature of gaining control in generative models which is regarded as a important missing feature in GAN.

Finally, the work of magnetograms predictions is also significant as we're the first to propose and formulate this problem, our research would be great value to astronomers as they could make physics interpretation on those solar cycles where they may lack data previously.

CHAPTER 6

FUTURE WORK

Our work provides a solid basis for each of the open problems, and they're also extendable regarding different aspects in the future work.

Background Subtraction

Pursuing the better and more powerful model is always the goal of deep learning research. Transformer based models have shown great potential in vision tasks, visual transformer has proved be the new state-of-the-art in many image related problems. The future direction for Background Subtraction task could be a specific designed transformer-based architecture for such video task.

Video Prediction with Action Control

The aim of gaining control in generative model has always been a important missing feature in Generative Adversarial Network (GAN), we proposed to use conformal map in our previous work to gain direction control' in video prediction task. But it indeed induced limitation that requires the latent space to obtain the same dimension as output space which reduces the representation capability of generative models.

Conformal mapping could be regarded as a special transform in the orthogonal group theory. Therefore, our future direction could be enforcing group properties into generative models to preserve rotation equivalence.

Magnetograms Prediction

The main purpose of this research lies in the lack of Michelson Doppler Imager (MDI) vector magnetograms for astronomical research purpose. Our previous solution proposed to trained a generative model on Helioseismic Magnetic Imager (HMI) data

and applied it on MDI data as both of the data obtain same structure and only differ in arcsecond resolution. The accuracy on MDI dataset is far from perfect and this motivates us to develop a more powerful model. Since these data are from two distinct instruments of different resolution, transfer learning between these two data domains maybe a good improvement direction of future work.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] Lars V. Ahlfors. *Conformal Invariants: Topics in Geometric Function Theory*. McGraw-Hill, New York, 1973.

[3] Mohammadreza Babaee, Duc Tung Dinh, and Gerhard Rigoll. A deep convolutional neural network for video sequence background subtraction. *Pattern Recognition*, 76:635–649, 2018.

[4] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. Stochastic variational video prediction. In *6th International Conference on Learning Representations*, 2018.

[5] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(12):2481–2495, 2017.

[6] Lukas Balles and Philipp Hennig. Dissecting adam: The sign, magnitude and variance of stochastic gradients. In *International Conference on Machine Learning*, pages 404–413. PMLR, 2018.

[7] Tommaso Banzato, Marek Wodzinski, Silvia Burti, Valentina Longhin Osti, Valentina Rossoni, Manfredo Atzori, and Alessandro Zotti. Automatic classification of canine thoracic radiographs using deep learning. *Scientific Reports*, 11:3964, 2021.

[8] Olivier Barnich and Marc Van Droogenbroeck. ViBe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing*, 20(6):1709–1724, 2011.

[9] Steven S. Beauchemin and John L. Barron. The computation of optical flow. *ACM Computing. Surveys.*, 27(3):433–467, 1995.

[10] Simone Bianco, Gianluigi Ciocca, and Raimondo Schettini. How far can you get by combining change detection algorithms? In *International Conference on Image Analysis and Processing*, pages 96–107, 2017.

[11] Marc Braham, Sébastien Piérard, and Marc Van Droogenbroeck. Semantic background subtraction. In *IEEE International Conference on Image Processing*, pages 4552–4556, 2017.

[12] Marc Braham and Marc Van Droogenbroeck. Deep background subtraction with scene-specific convolutional neural networks. In *2016 International Conference on Systems, Signals and Image Processing*, pages 1–4, 2016.

[13] Sebastian Brutzer, Benjamin Höferlin, and Gunther Heidemann. Evaluation of background subtraction techniques for video surveillance. In *CVPR 2011*, pages 1937–1944. IEEE, 2011.

[14] Wonmin Byeon, Qin Wang, Rupesh Kumar Srivastava, and Petros Koumoutsakos. Contextvp: Fully context-aware video prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 753–769, 2018.

[15] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2017.

[16] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018.

[17] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[18] Yubin Deng, Chen Change Loy, and Xiaoou Tang. Aesthetic-driven image enhancement by adversarial learning. In *Proceedings of the ACM International Conference on Multimedia*, pages 870–878, 2018.

[19] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1182–1191, 2018.

[20] Emily L. Denton and Vighnesh Birodkar. Unsupervised learning of disentangled representations from video. In *Advances in Neural Information Processing Systems*, pages 4417–4426, 2017.

[21] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *5th International Conference on Learning Representations*, 2017.

[22] Jeffrey Donahue, Lisa A. Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *2015 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.

[23] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martín Arjovsky, Olivier Mastropietro, and Aaron C. Courville. Adversarially learned inference. In *5th International Conference on Learning Representations*, 2017.

[24] Fida El Baf, Thierry Bouwmans, and Bertrand Vachon. Foreground detection using the choquet integral. In *9th International Workshop on Image Analysis for Multimedia Interactive Services*, pages 187–190, 2008.

[25] Thorsten Falk, Dominic Mai, Robert Bensch, Ã–zgÃ¼n Ã‡iÃ§ek, Ahmed Abdulkadir, Yassine Marrakchi, Anton BÃ¶hm, Jan Deubner, Zoe Jaeckel, Katharina Seiwald, Oleksandr Dovzhenko, Olaf Tietz, Cristina Dal Bosco, Sean Walsh, Deniz Saltukoglu, Tuan Leng Tay, Marco Prinz, Klaus Palme, Matias Simons, and Olaf Ronneberger. U-Net: deep learning for cell counting, detection, and morphometry. *Nature Methods*, 16:67–70, 12 2018.

[26] Jiashi Feng, Huan Xu, and Shuicheng Yan. Online robust PCA via stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 404–412, 2013.

[27] Chelsea Finn, Ian J. Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in Neural Information Processing Systems*, pages 64–72, 2016.

[28] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3146–3154, 2019.

[29] Francis Galton. Regression towards mediocrity in hereditary stature. *The Journal of the Anthropological Institute of Great Britain and Ireland*, 15:246–263, 1886.

[30] Richard Galvez, David F Fouhey, Meng Jin, Alexandre Szenicer, Andrés Muñoz-Jaramillo, Mark CM Cheung, Paul J Wright, Monica G Bobra, Yang Liu, James Mason, et al. A machine-learning data set prepared from the NASA Solar Dynamics Observatory mission. *The Astrophysical Journal Supplement Series*, 242:7, 2019.

[31] R. Gonzalez. *Digital Image Processing.* Pearson Hall, 3 edition, 2008.

[32] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[33] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[34] Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, 2013.

[35] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of Wasserstein GANs. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017.

[36] Eric Hayman and Jan-Olof Eklundh. Statistical background subtraction for a mobile observer. In *9th IEEE International Conference on Computer Vision*, pages 67–74, 2003.

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[38] Ruibing Hou, Hong Chang, Bingpeng Ma, and Xilin Chen. Video prediction with bidirectional constraint network. In *Proceedings of the 14th IEEE International Conference on Automatic Face & Gesture Recognition*, pages 1–8, 2019.

[39] D. C. Howell. *Fundamental Statistics for the Behavioral Sciences.* Wadsworth Cengage Learning, 2010.

[40] Zhihang Hu, Turki Turki, Nhathai Phan, and Jason T. L. Wang. A 3D atrous convolutional long short-term memory network for background subtraction. *IEEE Access*, 6:43450–43459, 2018.

[41] Zhihang Hu, Turki Turki, and Jason T. L. Wang. Generative adversarial networks for stochastic video prediction with action control. *IEEE Access*, 8:63336–63348, 2020.

[42] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017.

[43] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, 2013.

[44] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European Conference on Computer Vision*, pages 694–711, 2016.

[45] Taeyoung Kim, Eunsu Park, Harim Lee, Yong-Jae Moon, Sung-Ho Bae, Daye Lim, Soojeong Jang, Lokwon Kim, Il-Hyun Cho, Myungjin Choi, et al. Solar farside magnetograms from deep learning analysis of STEREO/EUVI data. *Nature Astronomy*, 3(5):397–400, 2019.

[46] Diederik P. Kingma. Fast gradient-based inference with continuous latent variable models in auxiliary form. *arXiv preprint arXiv:1306.0733*, 2013.

[47] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*, 2015.

[48] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[49] Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. In Michael A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 255–258. MIT Press, Cambridge, MA, USA, 1998.

[50] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, May 2015.

[51] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Handwritten digit recognition with a back-propagation network. In David S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 396–404, 1989.

[52] Alex X Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

[53] Dar-Shyang Lee. Effective Gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):827–832, 2005.

[54] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Flow-grounded spatial-temporal video prediction from still images. In *Proceedings of the 15th European Conference on Computer Vision, Part IX*, pages 609–625, 2018.

[55] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision*, pages 2999–3007, 2017.

[56] C. Liu, N. Deng, J. T. L. Wang, and H. Wang. Predicting solar flares using SDO/HMI vector magnetic data products and the random forest algorithm. *The Astrophysical Journal*, 843:104, July 2017.

[57] H. Liu, C. Liu, J. T. L. Wang, and H. Wang. Predicting solar flares using a long short-term memory network. *The Astrophysical Journal*, 877(121), 2019.

[58] Hao Liu, Chang Liu, Jason T. L. Wang, and Haimin Wang. Predicting solar flares using a long short-term memory network. *The Astrophysical Journal*, 877(2):121, June 2019.

[59] Hao Liu, Chang Liu, Jason T. L. Wang, and Haimin Wang. Predicting coronal mass ejections using SDO/HMI vector magnetic data products and recurrent neural networks. *The Astrophysical Journal*, 890(1):12, February 2020.

[60] Jiajia Liu, Yimin Wang, Xin Huang, Marianna B. Korsos, Ye Jiang, Yuming Wang, and Robert Erdelyi. Reliability of AI-generated magnetograms from only EUV images. *Nature Astronomy*, 5:108–110, 2021.

[61] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[62] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016.

[63] Michaël Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. In *4th International Conference on Learning Representations*, 2016.

[64] Marry L. McHugh. Interrater reliability: the kappa statistic. *Biochemia Medica*, 22:276–282, 2012.

[65] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge? *arXiv preprint arXiv:1801.04406*, 2018.

[66] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *11th Annual Conference of the International Speech Communication Association*, pages 1045–1048, 2010.

[67] Thomas B Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2-3):90–126, 2006.

[68] K. Nomizu and S. Sasaki. *Affine Differential Geometry*. Cambridge University Press, 1994.

[69] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L. Lewis, and Satinder P. Singh. Action-conditional video prediction using deep networks in Atari games. In *Advances in Neural Information Processing Systems*, pages 2863–2871, 2015.

[70] Marc Oliu, Javier Selva, and Sergio Escalera. Folded recurrent neural networks for future video prediction. In *Proceedings of the 15th European Conference on Computer Vision, Part XIV*, pages 745–761, 2018.

[71] Nuria M Oliver, Barbara Rosario, and Alex P Pentland. A Bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.

[72] K Pearson. VII. note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58(347-352):240–242, jan 1895.

[73] NhatHai Phan, Dejing Dou, Brigitte Piniewski, and David Kil. A deep learning approach for human behavior prediction with explanations in health social networks: social restricted boltzmann machine (SRBM+). *Social Network Analysis and Mining*, 6(1):79:1–79:14, 2016.

[74] NhatHai Phan, Yue Wang, Xintao Wu, and Dejing Dou. Differential privacy preservation for deep auto-encoders: an application of human behavior prediction. In *30th AAAI Conference on Artificial Intelligence*, pages 1309–1316, 2016.

[75] NhatHai Phan, Xintao Wu, and Dejing Dou. Preserving differential privacy in convolutional deep belief networks. *Machine Learning*, 106(9-10):1681–1704, 2017.

[76] Massimo Piccardi. Background subtraction techniques: a review. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, volume 4, pages 3099–3104. IEEE, 2004.

[77] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. In *Advances in Neural Information Processing Systems*, pages 2352–2360, 2016.

[78] MarcAurelio Ranzato, Arthur Szlam, Joan Bruna, Michael Mathieu, Ronan Collobert, and Sumit Chopra. Video (language) modeling: A baseline for generative models of natural videos. *arXiv preprint arXiv:1412.6604*, 2014.

[79] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015.

[80] P. H. Scherrer, R. S. Bogart, R. I. Bush, J. T. Hoeksema, A. G. Kosovichev, J. Schou, W. Rosenberg, L. Springer, T. D. Tarbell, A. Title, C. J. Wolfson, I. Zayer, and MDI Engineering Team. The Solar Oscillations Investigation - Michelson Doppler Imager. *Solar Physics*, 162(1-2):129–188, December 1995.

[81] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: A local SVM approach. In *Proceedings of the 17th International Conference on Pattern Recognition*, pages 32–36, 2004.

[82] A Sen and M Srivastava. *Regression Analysis*. Springer-Verlag New York, 1990.

[83] Moein Shakeri and Hong Zhang. COROLA: A sequential solution to moving object detection using low-rank approximation. *Computer Vision and Image Understanding*, 146:27–39, 2016.

[84] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-Chun Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, pages 802–810, 2015.

[85] Mohamad Hoseyn Sigari, Naser Mozayani, and H Pourreza. Fuzzy running average and fuzzy background subtraction: concepts and application. *International Journal of Computer Science and Network Security*, 8(2):138–143, 2008.

[86] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[87] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng. Parsing natural scenes and natural language with recursive neural networks. In *28th International Conference on Machine Learning*, pages 129–136, 2011.

[88] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[89] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using LSTMs. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 843–852, 2015.

[90] Pierre-Luc St-Charles, Guillaume-Alexandre Bilodeau, and Robert Bergevin. SuBSENSE: A universal change detection method with local adaptive sensitivity. *IEEE Transactions on Image Processing*, 24(1):359–373, 2015.

[91] Chris Stauffer and W. Eric L. Grimson. Adaptive background mixture models for real-time tracking. In *1999 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2246–2252, 1999.

[92] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. MoCoGAN: Decomposing motion and content for video generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1526–1535, 2018.

[93] Subhashini Venugopalan. Sequence to sequence - video to text. In *2015 IEEE International Conference on Computer Vision*, pages 4534–4542, 2015.

[94] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. In *5th International Conference on Learning Representations*, 2017.

[95] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *Proceedings of the European Conference on Computer Vision*, pages 835–851, 2016.

[96] Yi Wang, Pierre-Marc Jodoin, Fatih Porikli, Janusz Konrad, Yannick Benezeth, and Prakash Ishwar. CDnet 2014: An expanded change detection benchmark dataset. In *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 387–394, 2014.

[97] Yi Wang, Zhiming Luo, and Pierre-Marc Jodoin. Interactive deep learning method for segmenting moving objects. *Pattern Recognition Letters*, 96:66–75, 2017.

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

[99] Huan Xu, Constantine Caramanis, and Sujay Sanghavi. Robust PCA via outlier pursuit. In *Advances in Neural Information Processing Systems*, pages 2496–2504, 2010.

[100] Mighten C. Yip, Mercedes M. Gonzalez, Christopher R. Valenta, Matthew J. M. Rowan, and Craig R. Forest. Deep learning-based real-time detection of neurons in brain slices for in vitro physiology. *Scientific Reports*, 11:6065, 2021.

[101] Matthew D Zeiler, Graham W Taylor, and Rob Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *2011 IEEE International Conference on Computer Vision*, pages 2018–2025, 2011.

[102] Hongxun Zhang and De Xu. Fusing color and texture features for background model. *Fuzzy Systems and Knowledge Discovery*, pages 887–893, 2006.

[103] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10076–10085, 2020.

[104] Ce Zheng, Xiaolin Xie, Zhilei Wang, Wen Li, Jili Chen, Tong Qiao, Zhuyun Qian, Hui Liu, Jianheng Liang, and Xu Chen. Development and validation of deep learning algorithms for automated eye laterality detection with anterior segment photography. *Scientific Reports*, 11:586, 2021.

[105] Xiaowei Zhou, Can Yang, and Weichuan Yu. Moving object detection by detecting contiguous outliers in the low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):597–610, 2013.

[106] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2242–2251, 2017.

[107] Zoran Zivkovic and Ferdinand van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7):773–780, 2006.

[108] Fangyu Zou, Li Shen, Zequn Jie, Weizhong Zhang, and Wei Liu. A sufficient condition for convergences of adam and rmsprop. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11127–11135, 2019.