Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

R&D Project

Survey on Video-based Anticipation for Anomaly Detection

Anna Rose Johny

Submitted to Hochschule Bonn-Rhein-Sieg,
Department of Computer Science
in partial fulfillment of the requirements for the degree
of Master of Science in Autonomous Systems

Supervised by

Prof. Dr. Paul G. Plöger
M. Sc. Santosh Thoduka

January 2021

I, the undersigned below, declare that this work has not previously been submitted to this or any other university and that it is, unless otherwise stated, entirely my own work.

Date

Anna Rose Johny

Abstract

The ability to know about the future in advance is important in autonomous systems to avoid collisions. Anticipation is the process of obtaining future outcomes (example: trajectory, action, videos) based on past observations. Various anticipation methods find application in fields such as autonomous robots, self-driving systems, human-robot interaction. This research work provides a comprehensive survey of various anticipation methods that can be used for anomaly detection tasks. Anomaly detection is the process of detecting deviations of outcomes from nominal or expected outcomes. The existing surveys are application oriented and do not provide a complete evaluation of the methods, metrics, datasets used for anticipation in the field of autonomous systems.

In this survey, the anticipation methods are classified into action anticipation, video prediction and trajectory prediction. The datasets and the metrics used for these anticipation approaches are also evaluated in this survey. For analysis of different approaches, a comparison table is provided that compares various approaches with contributions and deficits of each approach. This survey helps the manufactures of autonomous systems to identify the best approach in anticipation and their applicability for anomaly detection tasks.

Keywords: Anticipation, Anomaly detection

Acknowledgements

I would like to thank my first supervisor, **Prof. Dr. Paul. G. Plöger** for his support and providing suggestion during the completion of this project. I would like to express my deepest gratitude to **M. Sc. Santosh Thoduka** for his support, motivation and guidance throughout the project. I thank my supervisors for continuous feedback and motivation throughout the project. Thank you for providing me with valuable suggestions during each part of the project.

I thank all my classmates for their support, motivation, and help to complete this project. I am grateful to all my friends for their valuable support throughout this project. Finally, I would like to thank my family for support and endless encouragement.

Contents

List of Abbreviations

ADE	Average Displacement Error
AGV	Autonomous Ground Vehicles
CNN	Convolutional Neural Network
DPG	Disentangled Propagation-Generation
EDM	Euclidean Distance Matrix
FDE	Final Displacement Error
GAN	Generative Adversarial Network
GPR	Gaussian Process Regression
GRUs	Gated Recurrent Units
JAAD	Joint Attention for Autonomous Driving
LSTM	Long Short-Term memory
MAE	Mean Absolute Error
MSE	Mean Squared Error
NGSIM	Next Generation Simulation
PSNR	Peak signal-to-noise ratio
RBF	Radial Basis Function
RED	Reinforced Encoder-Decoder
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
SAVP	Stochastic Adversarial Video Prediction
SSIM	Structural Similarity Index
STA	Spatiotemporal Accumulator

STAE	Spatio-Temporal AutoEncoder
TRN	Temporal Recurrent Network
VAE	Variational Auto Encoder
VAN	Visual Analogy Network

List of Figures

List of Tables

1 Introduction

The autonomous system should pose the ability to work by its own. To achieve this ability, the autonomous systems should know their surrounding in advance for making collision-free travel and also increase their safety. An anticipation is an approach by which the future surroundings can be known in advance. Anticipation is the process in which future outcomes such as trajectory, action and videos are predicted based on the previous observations. The anticipation finds applications in fields such as human-robot interaction [19][71], autonomous robots [55], self-driving systems [1], video surveillance. Anomaly detection is the process in which deviations are detected from nominal or expected outcomes. Anomaly detection can be used during preprocessing of the data or to find anomalies in the predicted data. Anomaly detection finds application in fields such as video surveillance, security, finance, autonomous robot. Most of the anticipation applications use image, video or trajectory information as the input and the output is a type of prediction that can be action anticipation, video prediction, trajectory prediction.

The objective of this research is to provide a comparison of different anticipation methods and finding their applicability for anomaly detection. In this survey, the term ”Anticipation” is used more because anticipation is more appropriate for anomaly detection because we are talking about nominal future. The survey report first analyses various anticipation methods and then identifies how to apply those methods for anomaly detection. In this survey, the types of anticipation considered are action anticipation, trajectory prediction and video prediction. After this classification tables are provided based on initial classifications mentioned earlier. Finally, a comparison of the methods, drawbacks, contributions of various papers is provided.

The basic idea of the survey is depicted in Figure 1.1. The anticipation model is trained only based on nominal data which allows us to predict future nominal behaviours based on past behaviours. By comparing observed and nominal behaviour anomalies can be detected. The anticipation network is provided with the observed data obtained from the robot’s sensors. This input to the anticipation model can be a sequence of images, videos or trajectory. The anticipation model is a prediction model made of various learning approaches such as feedforward architectures, recurrent models, gaussian models. The output from the anticipation model is a predicted video frame, predicted trajectory or anticipated action. This predicted nominal data are compared with observed data in order to determine the predicted value is nominal or anomalous, which is known as anomaly detection.

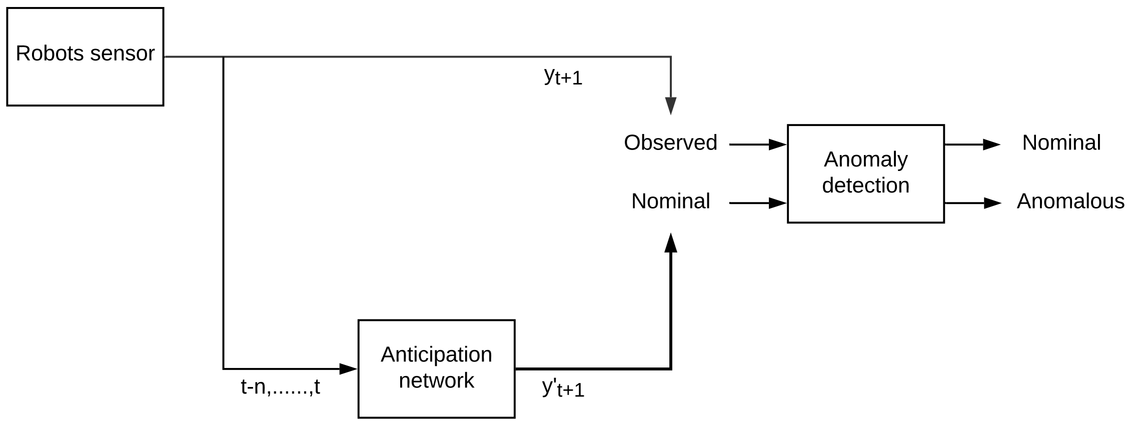


Figure 1.1: General idea of this survey

There is no comprehensive survey on anticipation methods for anomaly detection available. The existing surveys do not evaluate various anticipation methods such as trajectory prediction, action anticipation, future frame predictions, and their applicability for anomaly detection tasks. Absence of methods for classifying anticipation methods based on inputs, outputs, datasets, metrics and methods used. An existing survey presented in [60] on the prediction method provides basic classification on anticipation methods used for vision-based predictions. The anticipation method along with anomaly detection can be applied for the autonomous robots. This work benefits autonomous robots or self-driving car manufacturers to find the best method that incorporated both anticipation and anomaly detection. The survey on anticipation method is application based. An example of a survey on the type of anticipation is action prediction and recognition survey[37], which compares various approaches used for action anticipation and recognition. Survey on anomaly detection [10] is focused on methods and applications. These surveys do not provide a comparison based on inputs, datasets used. Surveys on anomaly detection do not cover anticipation types.

1.1 Motivation

Most of the autonomous systems need to operate in diverse environments without human intervention. In such cases, it would be a benefit for the systems to know their surroundings in advance. Anticipation based approaches help the autonomous systems to know their surroundings in advance to make collision-free travel. This is done by anticipating the future in advance. Also detecting anomalies in the anticipated output makes the safety even better.

Consider the example of self-driving car operating in real-world traffic. In this case, it is mandatory to evaluate the actions which are happening around in order to make collision-free travel. Here the anticipation can be used for predicting changing a lane, taking turns, pedestrians information and so on. It is a challenging task since the environments in which cars and autonomous robots operate are dynamic and the possible events are diverse. The anomalies in this case can be pedestrians crossing roads.

The main challenge in the estimation of the exact position of the pedestrians comes with the difficulty in collecting the image features required for tracking.

Another application is in the case of human-robot interaction, in which robots operate along with humans in the same field. This task requires high amounts of safety measures to be fulfilled. This anticipation is a kind of action prediction in which the robots will be able to observe the actions of humans for understanding their behaviour and predicting what they will do next. For the anticipation and anomaly detection, initially, the robot is trained with images obtained from the objects seen previously. During the testing phase, the new objects are detected which were not part of the trained data and are considered as anomalies. In this case, it can be a person walking through a corridor or an object placed on the floor can be an anomaly. The nominal conditions are anticipated and if we see new things that do not match our anticipation is an indication of an anomaly. This prediction helps the autonomous robot in operating with safety conditions by knowing the future prediction earlier.

1.2 Problem statement

The main intention of this Research and Development project is to conduct a comprehensive survey on anticipation methods for anomaly detection. The first task is to identify the learning approaches based on the application for anticipation. The main objective is to identify how these approaches can be applied for detecting anomalies by providing different datasets as inputs. Anomaly detection is important in the area of robotics because most of these systems require higher levels of autonomy. Identifying and correcting these anomalies are furthermore difficult. Another problem comes in choosing the datasets based on the application and the availability of datasets. All these problems are evaluated in this survey and the best approach is proposed finally based on the observations made.

1.3 Challenges and difficulties

The various challenges in this project are based on the methods chosen for the type of anticipation and anomaly detection. These challenges are further discussed in this chapter.

1. The uncertain and dynamically changing nature of the future is one of the key obstacles to be tackled by anticipation approaches [3]. The predictions are made based on past observations. If some new objects are entering this scenario, the predictions can go wrong.

2. In action anticipation tasks, the main challenge is to manage complex video data containing actions. Choosing the best action anticipation model that can identify the data based on the needed and irrelevant information is challenging because the input video data contains a large number of video frames, but all these video information may not be needed for a particular future sequence prediction [26].

3. The main challenge in the video prediction is to obtain exact predictions without blurriness [83]. The blurriness in predictions is due to the presence of Mean Squared Error (MSE) losses or directly accessing pixel frames for predictions. Another challenge in the video prediction is the dynamic

future and complex appearance of the predictions. For more realistic predictions, the pixel-wise data should be processed rather than directly accessing complex frames.

4. In the case of trajectory predictions, there should not be any delay in predictions. For example, a delay in future trajectory prediction can cause a collision in autonomous vehicles. The accurate and long-term prediction problems are difficult in predicting vehicle or pedestrian trajectories due to the constantly changing nature of the expected trajectory.

5. Unavailability of datasets: The anomaly detection is challenging because of the insufficient datasets for training a model. This is because the anomalous events are rare compared to the normal data in an anomaly detection dataset.

1.4 Contribution

The main contribution of this work includes a comprehensive survey on various methods used for various anticipation especially in terms of action anticipation, video prediction and trajectory prediction and finally evaluating their applicability for anomaly detection tasks. In this survey, the inputs considered are in the form of past video frames, robot actions, human pose. The expected output from the anticipation model is predicted video, predicted trajectory, anticipated action. The following chapters explains methods used, datasets, metrics used for action anticipation, video prediction and trajectory prediction.

1.5 Report outline

In this survey, various types of anticipations such as action anticipation, video prediction and trajectory prediction are considered. Chapter 2 provides an overview of various methods, datasets and evaluation metrics used for action anticipation and a comparison of these approaches are provided at the end. Chapter 3 provides details of video prediction by considering methods, datasets and evaluation metrics used. Chapter 4 provides an overview of methods, metrics and datasets used for trajectory prediction. The discussion of various approaches, methods, metrics used along with the application for anomaly detection is provided in Chapter 6. The final conclusion chapter provided contribution, lessons learned and future work.

2 Action Anticipation

Action anticipation refers to predict future actions provided input in the form of past videos or actions. Action anticipation and action recognition are comparable, but the difference is the method by which data is explored [37]. Figure 2.1 shows the difference between action anticipation and action recognition. As inferred from the figure, the major difference between these two terms is the time when predictions are made. In action recognition, the aim is to produce the whole action from video and then create a label for each category [37]. Action recognition finds application in normal scenarios in which predictions need not to be fast such as entertainment, video retrieval [37]. Whereas, in action anticipation, the aim is to detect the actions at earliest. Action anticipation is applied for scenarios that need current predictions such as autonomous systems, human-robot interaction. In this survey, motion prediction is also included under action anticipation, because motion prediction refers to predicting the sequence of actions. In the given scenario, action recognition identify the action, whereas action anticipation predicts future actions. For example, consider the case of autonomous vehicles. In this case, action anticipation can anticipate the future actions to avoid collisions. Whereas action recognition just analyses the video based on the current observations.

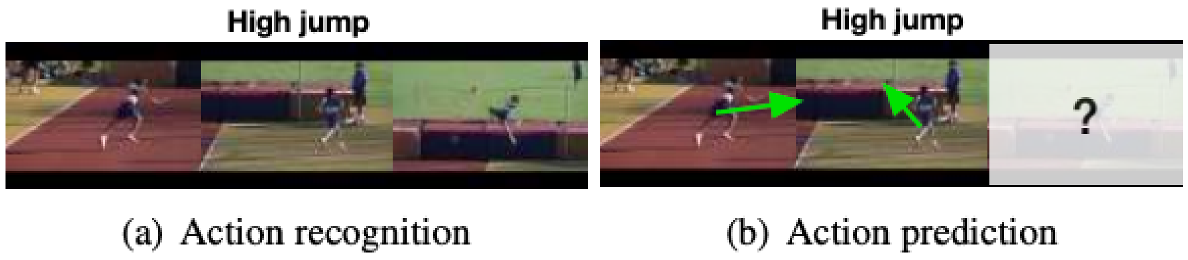


Figure 2.1: Difference between action recognition and action anticipation [37, p. 2]

In this survey, the term 'action anticipation' is used. Action anticipation finds application in fields such as self-driving systems [15], human-robot interaction [77], video-surveillance [35]. The next portions of this chapter discuss the various methods, datasets and metrics used for action anticipation. The methods used are classified into feedforward architectures, recurrent models and generative models. An overall summary of the suitable methods, datasets and the evaluation metrics are provided at the end.

2.1 Methods used

2.1.1 Feedforward Architectures

One of the paper uses the Convolutional autoencoder neural network for anticipating action, provided dynamic images as input [64]. A dynamic image is an image obtained by the concatenation of all the frames in the video frame into a single image [6]. The dynamic images are generated using future motion and expected appearance, and these generated frames are then passed through CNNs for anticipating future actions. This method is able to anticipate the actions by observing a few frames of images that contain actions [64]. In this approach, dynamic images are generated using CNNs and are named as dynamic CNN [64]. For increasing effectiveness of this method, the observed images are passed through static CNN [64]. In this approach, the observed images in RGB format is used for producing dynamic images and then used for anticipation tasks. Static CNN and dynamic CNN modules are present in this action anticipation network [64]. The idea of this anticipation is to anticipate future dynamic images given current dynamic images an input [64]. The model training is performed using the loss functions such as dynamic loss (finds the difference between predicted and ground-truth motion values being observed), classification loss (helps in producing dynamic images for anticipating actions) and static loss (the difference in appearance is evaluated by comparing the difference among ground-truth value and the predicted output) [64]. The method presented in this paper is able to produce future videos for the task of anticipation [64]. The method lacks the ability to apply for optical flow and also lack to predict dynamic images. The datasets used to test the method are UCF101, UT-Interaction, which contains human actions. Action anticipation by using video sequences [64]. To evaluate action anticipation performance, prediction accuracy is evaluated and showed 89.3% earlier predicted accuracy on UCF101 dataset [64]. The evaluation metrics depicted that the usage of hallucinated future dynamic images increased anticipation accuracy [64]. This method is able to anticipate only for shorter intervals of time or a particular dynamic image. [64]

The approach presented in [73] uses deep convolutional networks for the task of action recognition and thereby helps in action anticipation in videos. In this approach, spatial and temporal representations are analysed separately and then combined later and are known as a two-stream approach [73]. Spatial representations are used for performing action recognition from still video frames, whereas temporal information are used for action recognition of motion images [73]. This paper uses ImageNet dataset [5] (image classification dataset) for pre-training, thereby helps in identifying the spatio-temporal relations from the data being provided. The input video is divided into spatio-temporal, which is then combined in the end for action recognition in videos [73]. The method is evaluated by using UCF101 dataset [74], which showed an average accuracy. As mentioned earlier, the spatio-temporal observations separately, and is combined using softmax scores, otherwise, there are issues with overfitting of data. The main inference from the method is that this method shows training on stacked frames are better than training on raw frames of images [73]. The deficit with this approach is that the usage of CNNs for action anticipation does not consider trajectories and also suitable for smaller levels of anticipation.

For long term and efficient action anticipation, one time-conditioned method can be used [35]. This approach solves the problem associated with recurrent models for action anticipation, such as accumulated errors for long-term predictions. The time-conditioned action anticipation architecture is depicted in Figure 2.2. The time-conditioned method architecture consists of two parts such as attended temporal feature (for initial or immediate anticipation) and time-conditioned skip connection (for final anticipation) [35]. Initially, time representation and action classes are combined and then provided to the time-conditioned module, which is added along with the features for producing final anticipation. For the evaluation of the method, this method uses Epic-Kitchen and 50Salads dataset [35]. Each of these dataset consists of video frames of various length and can be utilized for action anticipation tasks. This method is able to anticipate actions after the 60s of observation [35]. This method solves the issue with CNN and RNN for action anticipation by reducing the accumulation errors and increasing the accuracy of prediction. The information about the last observed frames is clearly evaluated using time-conditioned skip connection [35]. Usage of skip connection improves the performance of the approach and making it available for short-term and long-term predictions. Another advantage of using this approach is the requirement of only the last observation for anticipation [35]. This method is able to anticipate the action of various diverse scenarios and making correct predictions even for longer duration [35].

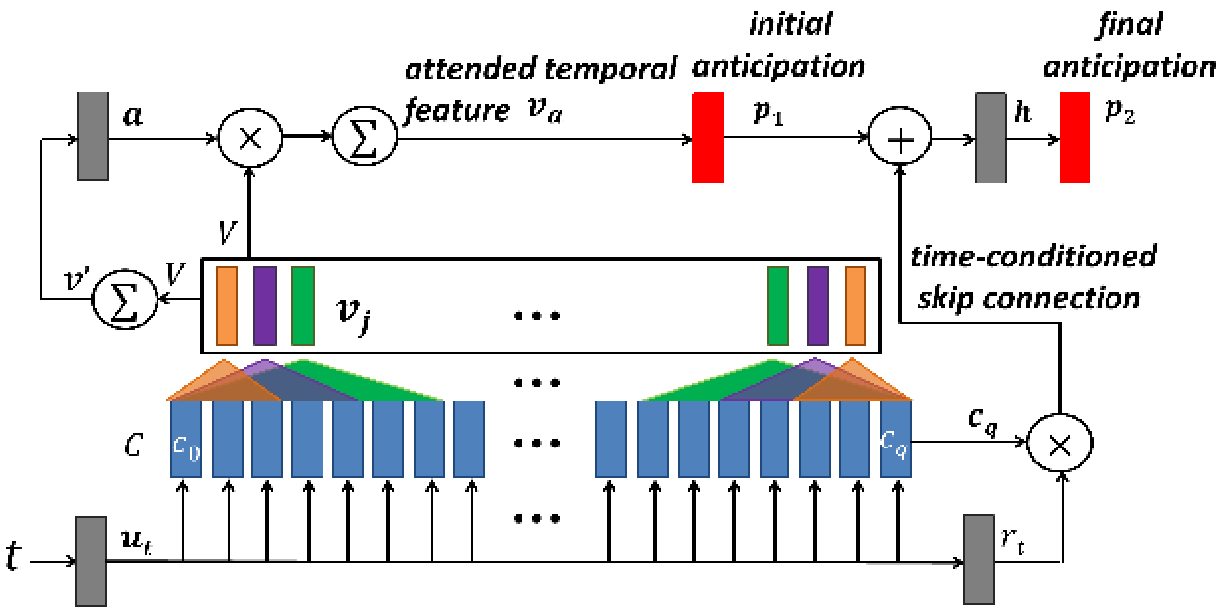


Figure 2.2: Architecture of time-conditioned action anticipation [35, p. 9927]

For the Autonomous Ground Vehicles (AGV), the requirement of understanding about its surroundings especially the pedestrians in the scenario is important, for this application one of the approach uses Spatio-Temporal DenseNet model [68]. The input provided is in the form of image sequences obtained using an RGB monocular camera [68]. The paper presents a real-time framework that is capable of detecting, tracking and finally predicting pedestrians actions using tracking-by-detection technique [68].

These spatio-temporal convolutional networks (3D) are more suitable than RNNs for image sequences as input. The usage of DenseNet (Dense Convolutional Network) architecture reduces the GPU requirements for spatio-temporal convolutional networks, which hinders the real-time applications feasible [68]. To predict the actions of pedestrians in traffic scene, initially the k-image sequences should be fed into the network, which is done using tracking-by-detection method [68] and is best suitable for this application because of providing an accurate estimation of image sequence. The basic architecture of this model consists of multi-pedestrian tracking which is implemented using tracking by detection method and then spatio-temporal DenseNet model [68]. The architecture of DenseNet consists of a sequence of convolutional blocks made of Rectified Linear Unit (ReLU)-3D Convolution [68]. To control the number of input features, dense block layer and bottleneck layers are used in this approach [68]. Joint Attention for Autonomous Driving (JAAD)[61] dataset is used for training the model. The spatio-temporal DenseNet architecture is suitable for predicting the action of pedestrian in an urban environment and it is suitable for autonomous driving [68].

2.1.2 Recurrent Models

An extension of feedforward architectures is RNN, consist of a hidden state that is capable of learning temporal representations of sequential data [62]. Early action anticipation in the case of detecting vehicles maneuver for the autonomous driving uses a combined RNN-LSTM network [27]. The input features are extracted from the input data using a preprocessing, which is provided to the RNN-LSTM based network for the task of anticipating the action [27]. The method also uses a smoothing function for noise suppression. The noise suppression is done using filters and correction function [27]. In this application, LSTMs are used because they require less time for prediction [27].

For online action anticipation and detecting the future frames earlier, one of the approaches uses Temporal Recurrent Network (TRN) [87]. The model differs from the previously mentioned approach in the way of anticipating the actions simultaneously [87]. The working of TRN is similar to RNN, the sequence of video frames are provided as input and output are set of probabilities [87]. Another difference with RNN is the temporal decoder for anticipating future actions and then using this information to provide a more accurate online action detection and anticipation [87]. The main feature of online action anticipation model is to detect and process the data soon after it appears [87]. The TRN cell is made up of temporal decoder, Spatiotemporal Accumulator (STA) and future gate [87]. The internal information of the temporal decoder and STA is controlled using the LSTM network. The LSTM is used to learn features from videos and then predicting future actions [87]. The temporal decoder helps in predicting future actions and the hidden states by a sequential network for which the input at the first step is zero and for the next times, the predicted action scores are added [87]. The future gate receives information about the hidden state from the decoder and use this information for representing future actions [87]. The hidden state information along with the feature vector taken from the image and the predicted feature from the future gate is provided to STA [87]. The method uses STA for obtaining historical observations, current information and also information about predicted frames, using this for estimating action occurring in the current frame [87]. A combination of accumulator and decoder loss is included to ensure proper working of the action prediction module [87]. The model implemented can be called an online module, because this uses only the predicted future motion information during testing phase [87]. This method is evaluated using the datasets such as TVSeries, HDD, THUMOS'14 [87]. The evaluation metrics used are mean average Precision (mAP) and calibrated average Precision (cAP) [87]. This method is suitable for early action detection and then predicting future action with online detection method [87].

For action anticipation, one of the paper uses a feature mapping RNNs [72]. The machine learning principles used in this approach are Radial Basis Function (RBF), parameter sharing and adversarial training [72]. The model is implemented to predict actions in videos [72]. The RNN network is used to map the feature vector at time t to the prediction of features for time $t+1$. In this case, some regularization is done to avoid over-fitting of the data. In order to deal with complex data, parameter sharing is used [72]. RNNs are used to share parameters in temporal dimension [72]. To analyse complex data, a mapping layer is included inside RNN, which is processed using RBF kernels for analysing feature vector. The overall architecture of feature mapping RNN is showed in Figure 2.3. Initially, the RGB image $I(t)$ taken from the video data is provided to the deep CNN to extract the features at the time t and is denoted as x_t [72]. This feature vector is then divided into small parts of equal length and is then provided to LSTM, which is the prediction network [72]. Finally, these smaller predicted segments are combined to form the higher quality predicted sequences for the time $\hat{x}_{t+k}$. Using RBF kernel, increases prediction accuracy also the classifier performance [72]. The action anticipation is performed by observing the starting video frames from videos provided. The normal RNN for feature generation uses the sequence of feature vectors observed to predict the future frames and the hidden features [72]. This approach uses feature mapping RNN with RBF kernel to improve the precision accuracy [72]. The MSE loss can cause blurry predictions, to avoid that a combination of adversarial loss and L2 loss are used [72].

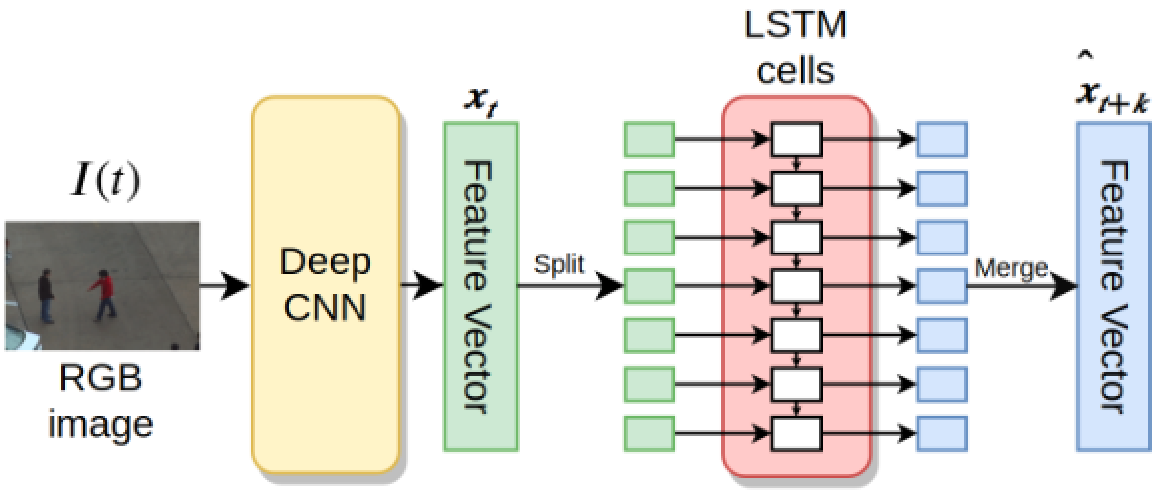


Figure 2.3: Action anticipation using feature mapping RNN [72, p. 2]

Anticipating action sequences for human-robot cooperation, paper [71] uses the encoder-decoder recurrent network. The approach presented in this paper uses a feature selection method to predict action sequences of variable length and also multiple future action sequences [71]. The first module of the prediction network consists of contextual information encoder, which helps in compressing the past observations using LSTM [71]. Followed by this layer is a fully connected embedding layer, which is used for regularizing the model [71]. Next module is the decoder, which is formed by the internal state of the previous encoder LSTM and is accountable for generating the action sequences [71]. A softmax layer is used for normalizing the data obtained from the internal state of the decoder and is in the form of discrete distribution [71]. The output from this decoder is then provided as the input to the next decoder and continues [71]. The model training uses a sequential cross entropy calculated by summation of the cross entropy along the cost required for prediction [71]. The decoder is capable of predicting action sequences of differing lengths, but this is computationally complicated [71]. This problem is solved using beam search encoder-decoder that stores only the best sequences [71]. This approach is application specific and is applied for human-robot collaboration. The model is initially provided with a set of robot actions, then the robot selects the action sequence that maximizes the probability and finally the action performed by the robot in the final step is evaluated and is used as predicted action sequence [71]. The dataset used for evaluation is CAD120 RGB-D, and contains various actions such as pouring, eating [71]. The performance is assessed using F1-score defined as the ratio of true positive given sum of true positive, false positive and false negative [71]. The method showed higher accuracy in predicting longer sequences and also suitable for predicting action sequences of differing length[71].

For pedestrian action anticipation, stacked RNN approach presented in [62] can be used. This approach is similar to pedestrian action prediction using DenseNet presented in [68]. For predicting pedestrian behavior or action, the time should be considered when autonomous vehicle and pedestrian come in contact (crossing roads) [62]. The stacked RNN architecture consists of Gated Recurrent Units (GRUs) stacked by concatenation of features and they are connected in a sequential way (output of one GRUs is provided as input for next GRUs) [62]. The complexity of this network increases with the number of features and each of these input features contain m number of observations (m is the time required) [62]. The appearance and surrounding features of pedestrians are processed using CNNs, from which pose of person and speed of the vehicle are fed into GRUs [62]. The datasets used for anticipating the actions of the pedestrian are JAAD and Pedestrian Intention Estimation (PIE) dataset [62]. For the longer observations, the performance of this approach reduces due to presence of noise accumulation. Also, the performance of this approach reduces with the complex features added to higher levels of stacked RNN[62]. This approach uses complex features at a lower level and simpler features at a higher level to improve accuracy of the method [62]. This approach is suitable for pedestrian action anticipation, but with reduced anticipation accuracy due to difficulties with dynamically changing environment [62].

For early action anticipation, one of the approaches uses multi-stage LSTM [67]. The higher prediction accuracy and ability to anticipate even from shorter sequences of video makes this approach different from other early action anticipation approaches. This model is able to extract action-aware and context-aware features from videos [67]. This approach consists of two stages, and which the first stage consists of extracting context features from the RGB image and then these context information is combined with action-aware features for obtaining an idea about where the actions are occurring in various sequences [67]. This paper uses a loss function for early action anticipation, which should be the higher score to facilitate early predictions. This loss function consists of two modules such as true activity label (false negatives) and the label predicted by the model (false positives) [67]. The model architecture of the multi-stage recurrent model consists of a shared parameter block, which is a block shared by context and action-aware features [67]. Next block is a separate block for each stream and then train a model using cross-entropy loss function [67]. For reducing memory usage and computational complexity, this paper adopts to train only two-sub models instead of training the entire model [67]. The basic architecture consist of LSTM layers and fully connected layer from which class probabilities are obtained [67]. The datasets used are UCF101 and UT-Interaction dataset [67]. Usage of action-aware features helps in eliminating irrelevant information by choosing only the best actions required for anticipation. This method is suitable for early action anticipation by using context and action-aware features using multi-stage LSTM [67]. This method cannot be applied for complex scenarios whether the environment changes dynamically [67].

One of the paper uses a Reinforced Encoder-Decoder (RED) for action anticipation [23]. This approach utilizes the entire percept history to learn and then anticipate the sequence of future frames [23]. The basic architecture of the RED network consists of three modules such as extractor, encoder-decoder network, classification network [23]. Initially, the extractor module is used to extract information from video representation. For this task, the video is divided into small parts and is processed using a feature extractor. After the features have been extracted, the video is then fed to the encoder-decoder network, which detects the historical observations and helps in anticipating future actions [23]. Anticipated actions are classified into various categories using a classification network. The encoder-decoder network is made up of a sequence of LSTM network [23]. The input to this module is provided in the vector form of video at a particular instant of time. The last output from the encoder module is given to the decoder and provides outputs based on the values being evaluated. The action anticipation is provided by a target sequence, which is a sequence that is obtained after the input sequence provided to the encoder network [23]. This approach uses squared loss for training network [23]. The approach also uses a reward function to ensure the predictions are correct or not and also to ensure the time taken for anticipation [23]. This function is evaluated as the time decreases, the chances of making correct predictions increases [23]. For training this network, unlabeled samples can be used. There are two stages that evolved in the training, which is regression loss and overall function [23]. This method is evaluated using TVSeries, THUMOS'14, and TV-Human-Interaction datasets [23]. Anticipation performance is improved by the usage of the encoding sequence of history representations [23]. The RED method is suitable for correct and early anticipation applications.

2.1.3 Generative Models

To anticipate 3D human motions and to provide long-term predictions, GAN networks can be used. The approach used in [30] evaluates the spatio-temporal dimensions of the data, thereby helps in providing accurate predictions regarding the exact position. This paper refers to motion prediction rather than action anticipation. The architecture of spatio-temporal inpainting GAN consists of generator and discriminator modules as shown in Figure 2.4 [30]. Inpainting refers to providing complete image or motion data from an incomplete sequence provided as the input [30]. The generator module is initially provided with the masked human motion sequence (S^m) as the input [30]. The difficulty in processing the masked human motions directly is solved using a generator U-block, which is inserted between frame encoder and frame decoder in the main architecture of the generator framework [30]. The generator blocks are made up of CNNs, which helps in processing both spatial and temporal dimensions [30]. Initially, the frame encoder is provided with occluded sequences and is passed to the generator U-block to produce new sequences [30]. These generated new sequences are then passed to the frame decoder which maps these sequences back to the output sequence.

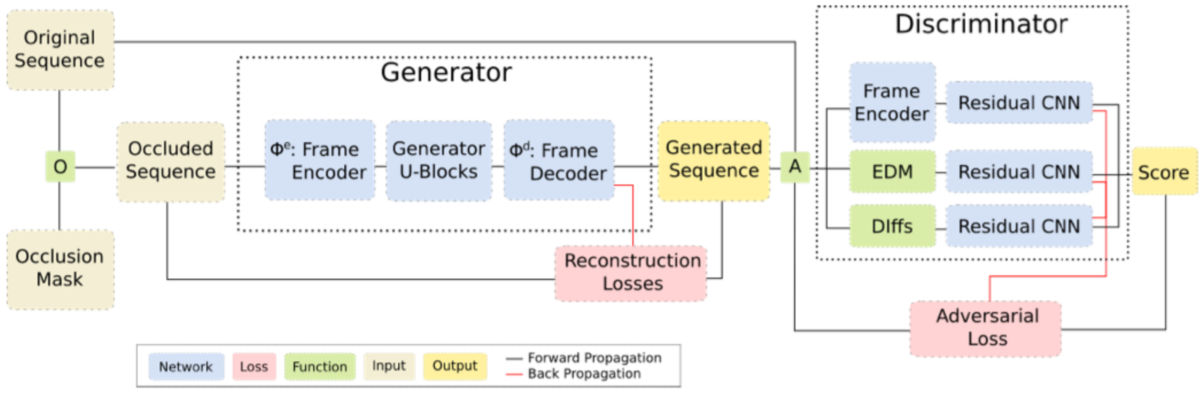
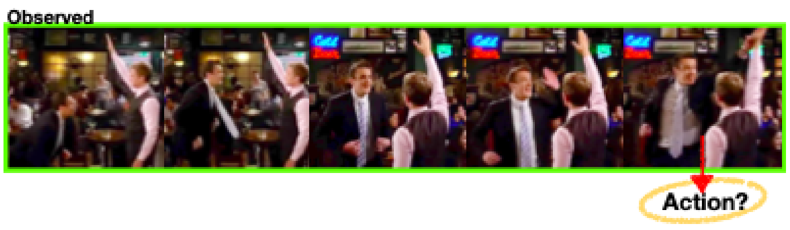


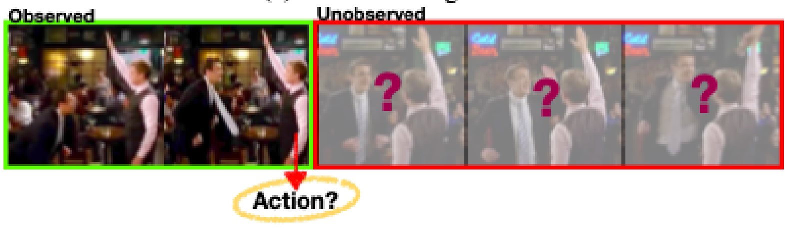
Figure 2.4: Architecture of Spatio-temporal Motion Inpainting GAN (STMI-GAN) [30, p. 3]

The discriminator module is split into various segments such as base, Euclidean Distance Matrix (EDM), and motion discriminator for capturing complex human motions [30]. The base discriminator works similar to the frame encoder in the generator block, but the sequences are generated with independent parameters, using this increases the performance of the model. The EDM discriminator is used to estimating the human-like characteristics of the generated sequences using EDM [30]. Finally, the motion discriminator concatenates the other two discriminators and is made up of residual CNN for feature extraction [30]. This approach uses two losses for training the network such as reconstruction loss (helps generator to preserve the information) and GAN loss (helps to inpaint sequences) [30]. The dataset used is Human3.6 M for evaluation, from which it is observed that the predictions made were correct but the speed of the predicted motion was inaccurate [30]. This approach is able to predict human motions using historical skeleton poses using GAN architecture [30]. The approach is able to predict accurate realistic human positions but fails in providing correct predictions within the time provided [30].

For early action anticipation, GAN can be used [20]. This anticipation approach can be used for anomaly detection by comparing the observed frames and predicted frames [20]. If the predicted frames matches with the observed ones, then it is non-anomalous otherwise anomalous event [20]. This method also captures both temporal and visual representations of the data provided [20]. Figure 2.5 explains basic concept of action anticipation that uses small portions of observed frames for predicting future actions along with anomaly detection.



(a) Action Recognition



(b) Typical Action Anticipation

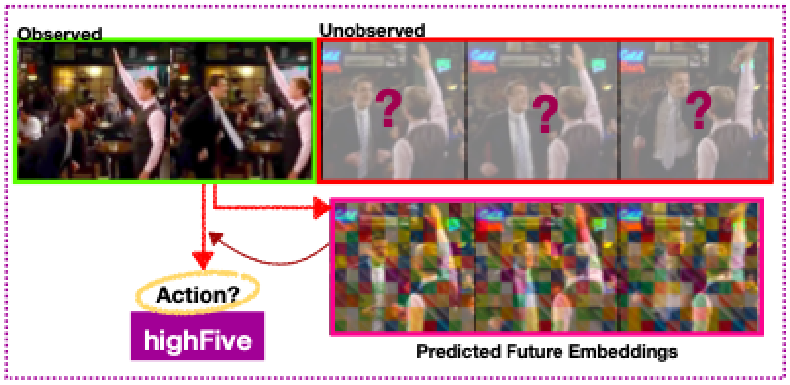


Figure 2.5: Prediction based on observed and unobserved frames [20, p. 5562]

As shown in Figure 2.6, this approach contains the two GAN model training simultaneously for future temporal and future visual representations [20]. The model is provided with the inputs in the form of video frames, from which visual and temporal representations are observed. The joint learning framework uses a feature extractor to extract data rather than processing raw frames [20]. The low-level feature are transferred into high-level features using attention mechanism, that merges the individual streams from the feature extractor [20]. This information is then provided to GAN modules, which then classifies the predicted frames into real or fake images [20]. The generator is made up of LSTM and is fed only with nominal data whereas the discriminator module is fed with nominal and fake data [20]. These inputs are provided to discriminator using LSTM and the output obtained is combined, passed through fully connected layers [20]. The evaluation datasets used are UCF101, UT-Interaction, TV Human Interaction and results showed that the anticipation model can predict the future actions despite provided fewer frames [20]. This approach is flexible with different dataset sizes and video lengths.

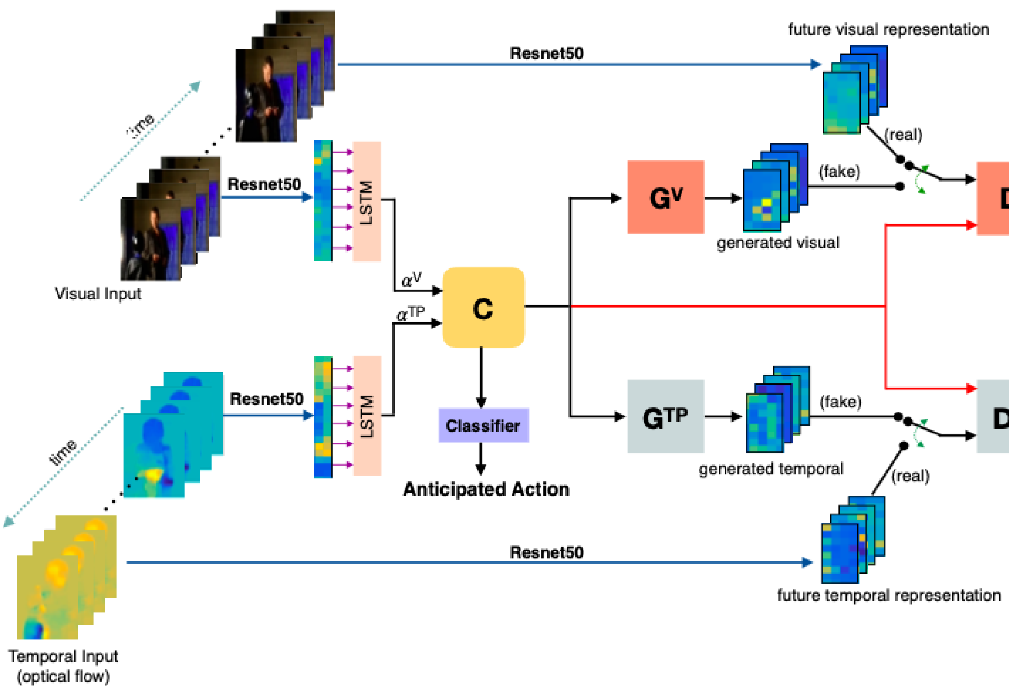


Figure 2.6: AA-GAN architecture [20, p. 5564]

2.2 Datasets

UCF101 dataset [74]

This is one of the largest datasets of action recognition containing trimmed videos. This dataset consists of 101 classes, that contain various human actions such as apply eye makeup, haircut, sky diving, rafting, and so on [74]. The dataset also contains videos from YouTube, which has challenges such as cluttered background, poor lighting conditions [74]. The early action anticipation tasks can be performed using the UCF101 dataset [72].

Epic-Kitchen dataset [12]

Epic-Kitchen is the largest dataset that contains videos related to human interaction with objects in cooking scenarios. This dataset consists of 39596 action segments with 432 sequences and 11.5M frames [12]. Action anticipation tasks are done using the Epic-Kitchen dataset.

50Salads dataset [35]

The 50Salads dataset contains 50 RGB-D video data of 25 people preparing two salads. This is an action recognition dataset but can be used during training for action anticipation.

UT-Interaction dataset [66]

This dataset can be used for anticipating actions [72], which contains 20 videos having a minimum length of 1 minute, with complicated dynamics. The videos are about various human interactions such as a hug, push, punch, and so on. Each of these videos contains various interactions and also different persons with different types of clothing. The data provided in this dataset is having a higher resolution.

TV Human interaction dataset [58]

The TV Human interaction dataset consists of 6776 video clips [58]. The videos are from various TV shows and are used for predicting actions such as a handshake, hug, high five, and so on. This dataset provides interaction among human beings, which can be used during action anticipation tasks.

THUMOS dataset [23]

THUMOS dataset consists of 45 million video frames for detecting and classifying actions [23]. This dataset is already preprocessed based on actions classified. The dataset consists of videos in form of text, images, and videos.

Joint Attention for Autonomous Driving (JAAD) dataset [61]

This dataset can be used for predicting pedestrian actions. The dataset contains 300 video clips with 5 to 15 seconds time duration [61]. The data is obtained using a high-resolution monocular RGB camera. This dataset contains information about pedestrian walking, looking, and standing. This dataset provides higher-level information about various actions and thereby can be used for action anticipation.

TVSeries dataset [23]

TVSeries dataset is one of the largest dataset for action detection and is used for action anticipation tasks in [23][87]. The dataset contains videos from six different with 30 action classes. The various action classes are drink, throw something, drive car, open door and so on [24].

2.3 Metrics

The performance of the anticipation models is evaluated using metrics. The most common metrics for action anticipation include anticipation accuracy, precision, recall, F1-Score.

Anticipation accuracy [35][72][20][23][71][81]

Accuracy can be defined as ”the ratio of correct predictions made and the total number of predictions” [60]. The accuracy evaluation is only based on correct predictions, which may not true for incorrect predictions. For this, along with accuracy, precision is also used for evaluating anticipation performance.

The formula for calculating anticipation accuracy is given as,

$$Anticipation\ accuracy = \frac{Correct\ predictions}{Correct\ predictions + Incorrect\ predictions} \tag{2.1}$$

Average Precision(AP) [23][68]

The average precision is calculated as ”the ratio between the negative and positive frame” [23]. It can also be calculated as calibrated AP (cAP) [23], mean AP (mAP) [35][23].

$$cAP = \frac{TP}{TP + FP/w}[23][87] \tag{2.2}$$

Where w is the ratio between negative and positive frames, TP is true positive, FP is false positive [23].

F1-score[64]

F1-score is calculated using the formula,

$$F1 = \frac{2.True - Positive}{2.True - Positive + False - Negative + False - Positive}[71] \tag{2.3}$$

Other evaluation metrics used for action anticipation are **recall** [35][72], **Area under curve(AUC)** [64], **averaged frame-wise accuracy** [81].

2.4 Summary

In this section, a summary of the various approaches used for action anticipation will be discussed. In case of action anticipation tasks, the anticipation model input can be a sequence of images or video. For action anticipation, most of the papers use recurrent models rather than feedforward architectures. Recurrent models are mostly used for action anticipation because of their flexibility in handling data of various length and type, but the computational cost of the recurrent models is higher due to the requirement of preprocessing steps to manage high dimensional data. Recurrent model-based approaches can be used for early action anticipation or long term predictions. Most of the recurrent model approaches are using LSTM network. In contrast to this, the feedforward architectures are able to predict future actions in one-time conditioned, thereby providing shorter processing time, but these approaches lack the ability to make dense anticipations. The feedforward architectures used are CNN, DenseNet. Some of the action anticipation approach focus on predicting the immediate future, which is suitable for single future anticipation applications such as human-robot interaction. For predicting future that involve multiple agents in a scene, just the immediate predictions are not necessary. To solve this recurrent models can be used for manipulating high-dimensional data. The approaches using recurrent models are most suitable for short term predictions and also having less accuracy compared to feedforward architectures. Long-term and more realistic predictions are given by generative models when compared to other approaches.

The datasets are common for most of these papers and are used based on application. The common datasets used are UCF101, UT-Interaction, Breakfast, Epic-kitchen, 50Salads and so on. The above-mentioned datasets are mostly action recognition datasets but used for action anticipation tasks. All of these datasets contain actions performed by a human in various scenarios.

The most common metrics used for evaluating the performance are accuracy, precision, recall, and F1-score. All these conclusions are tabulated below. Most of the methods are evaluated using the prediction accuracy, but this evaluation metrics is not suitable for data with variations in the amount. Another metric used for action anticipation model evaluation is precision. The metrics used for evaluation also varies based on the application. Table 2.1 provides the overall idea about these comparisons.

Reference	Method used	Dataset	Contribution	Drawback
Feedforward architectures				
Ke et al.[35]	One time-conditioned	Epic-Kitchen[12], 50Salads, Breakfast	Short and long-term prediction	Less efficient for dense anticipation
Rodriguez et al.[64]	Convolutional autoencoder network	UCF101, UT-Interaction	Predicting future motion representations for action anticipation	-
Simonyan et al.[73]	Deep convolutional network	ImageNet (training), UCF101 (evaluation)	Two-stream ConvNet architecture with spatio-temporal networks	Not suitable for trajectories
Miech et al.[54]	CNN based predictive and transition model	Epic-Kitchen, Breakfast	Anticipate human action seconds before they occur using predictive and transition model constructed using CNN architecture	-
Saleh et al.[68]	Spatio-temporal DenseNet	JAAD[60]	Action prediction of pedestrians from video-sequences	Application specific
Liu et al.[46]	SSNet (Scale Selection Network)	PKU-MMD dataset	Real-time 3D action anticipation with convolutional layers to get temporal dimensions	Less accuracy in predictions
Pirri et al.[59]	Action Forecasting Network (AFN)	UCF101, JHMBD	A model to anticipate current action and also foresee next action using ProtoNet architecture	Accuracy decreases for the samples selected between current and next action

Table 2.1: Summary of methods, datasets used for action anticipation

Reference	Method used	Dataset	Contribution	Drawback
Recurrent models				
Shi et al.[72]	Feature mapping RNN with RBF	UT-Interaction, UCF101	Model that can be trained with labeled and unlabeled video data	Over-fitting when RBF used with many kernels
Rasouli et al.[62]	Stacked RNN	JAAD[61]	Pedestrian action anticipation in video-sequences	Less accurate for complex features
Gammule et al.[21]	Neural memory network - LSTM	Breakfast, 50 Salads	Forecast sequence of actions	Only for shorter sequences
Xu et al.[87]	Temporal Recurrent Network (TRN)	THUMOS'14, TVSeries	Online action anticipation model for anticipating immediate future	Less prediction accuracy
Wang et al.[81]	3 layered RNN with LSTM	PKU-MMD dataset[45]	3D action anticipation from streaming videos for short video clips as input	Chances for less precision for a longer sequences as input[81]
Gao et al.[23]	Reinforced Encoder-Decoder (RED)	TVSeries, THUMOS'14, TV-Human-Interaction	Correct and early anticipation	-
Jain et al.[34]	Sensory fusion RNN-LSTM	Driving dataset	Model that is able to predict actions even with partial temporal context	Lower performance
Kong et al.[38]	Deep sequential context network (DeepSCN)	UCF101, Sports-1M	Shot-term action label prediction with higher speed of prediction	Less prediction accuracy, some predictions are wrong
Furnari et al.[18]	Rolling-unrolling LSTM	Epic-kitchen	Egocentric action anticipation using two LSTMs	-
Schydlo et al.[71]	Encoder-decoder RNN	CAD120 RGB-D motion dataset	Variable length action sequence prediction	Application specific
Aliakbarian et al.[67]	Multi-stage LSTM	UT-Interaction, UCF101	Higher prediction accuracy with very small video sequence	-

Table 2.1: Summary of methods, datasets used for action anticipation

Reference	Method used	Dataset	Contribution	Drawback
Combined models (Feedforward and recurrent models)				
Farha et al.[1]	CNN and RNN	Breakfast, 50Salads	Long term prediction	Error propagation in RNN, requirement of more parameters for CNN
Generative models				
Gammule et al.[20]	Action Anticipation GAN(AA-GAN)	UCF101, UT-Interaction, TV Human Interaction	Joint learning framework for early action anticipation	-
Wang et al.[80]	GAN(with generator CNN, LSTM)	UCF101, UT-Interaction	Early accurate action anticipation from partially observable videos	-
Hernandez et al.[30]	GAN	Human 3.6M	Long term realistic prediction	-

Table 2.1: Summary of methods, datasets used for action anticipation

3

Video Prediction

Video prediction is defined as predicting the future frames based on the inputs in the form of video sequences (previous frames) [57]. A collection of sequence of frames are known as video. Video prediction finds application in fields of autonomous driving, human-robot interaction, robot navigation systems [57]. One of the existing surveys presented in [57] for video prediction discusses various learning approaches, datasets used for video prediction. The challenging task in future video prediction is that the events are of a diverse nature. The main issue associated with video prediction is the chances of blurry predictions. The blurry predictions are due to accessing the pixel values directly for predictions. All these challenges and their solutions are addressed in the following parts of this chapter. For predicting the realistic pixel values in future frames, the model should possess the capability of capturing the motion changes and pixel-wise appearance, in order to provide the previous frames as the inputs for producing new frames.

3.1 Methods used

3.1.1 Feedforward Architectures

For higher quality video prediction one of the approaches presented in [22] uses a Disentangled Propagation-Generation (DPG) network as shown in Figure 3.1. The model input is the video sequence of the previous frame denoted by $x_{1:t-1}$ [22]. The model consists of two separate modules such as flow predictor (F) and occlusion inpainter (P) [22]. The flow predictor is provided with the history frame $x_{1:t-1}$, from which correlation between the features are evaluated and then predicts the future using optical flow [22]. The flow predictor is made up of convolutional encoder-decoder network and the output expected is a predicted frame $\tilde{x}_t$ [22]. The next module is an occlusion inpainter, which is also made of convolution encoder-decoder network. The input to the encoder that produces the latent feature representation is predicted frame $\tilde{x}_t$ from the flow predictor and the occlusion map $\tilde{m}_t$ [22]. The output from the encoder is then provided as the input to the decoder module, which helps in analysing the missing data. The two modules are trained separately to ensure higher accuracy of the predicted frame. To evaluate the predicted accuracy, Peak signal-to-noise ratio (PSNR) and Structural Similarity Index (SSIM) are used, and the higher values of these measures indicate better performance [22]. The datasets used for evaluation are KITTI and CalTech pedestrian. The frames are down-sampled and the first 10 frames are provided to the model and 11th frame is then predicted with higher accuracy [22].

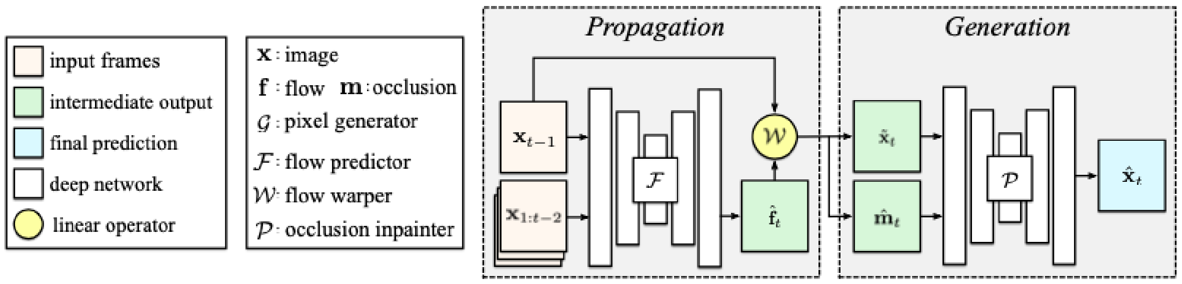


Figure 3.1: DPG architecture [22, p. 9007]

One of the approaches for video prediction and detecting anomalies in videos is to use Spatio-Temporal AutoEncoder (STAE) [92]. In this approach 3D convolutions are used instead of 2D convolutions, because the 2D convolutions can only get the temporal data and are suitable for action recognition tasks.The 3D convolutions are able to detect and predict the data in both spatial and temporal directions, which is much needed for video prediction tasks. The 3D convolution layer is formed by the concatenation of different continuous frames of temporal data. The formula for obtaining data of l^{th} layer and at position (x, y, z) of the 3D convolutional layer [92] is given as,

$$\alpha_l^{ixyz} = f(\sum_{k=1}^{K_l}\sum_{h=1}^{H_l}\sum_{w=1}^{W_l}\sum_{d=1}^{D_l}\theta_{l,i}^{khwd}\alpha_{l-1}^{k(x+h)(y+w)(z+d)} + \beta_l^i)[92] \tag{3.1}$$

Where H_l, K_l, W_l, D_l indicates the height, channel number, width, depth of the 3D kernel at l^th layer respectively [92]. The motion details are obtained by connecting D_l layer to the previous feature map layers [92]. The STAE architecture is constructed based on this 3D convolutional layer. Input to the STAE network is the video having different frames instead of a single image as in case of image recognition tasks [92]. In this approach, T (sliding window length) number of frames are stacked together and this hyper-cuboid is given as the input to autoencoder. Due to the lack of large training dataset for anomaly detection in videos, more inputs are generated using different transformations such as brightness changing, random cropping [92]. In the anomaly detection tasks, the speed of detection is an important criterion [92]. To maintain a constant speed of the objects, constant strides are used to sample the frames [92]. The 3D convolutional autoencoders architecture [92] is depicted in Figure 3.2. For the STAE network, the frame number is set to $T = 16$, each frame is rescaled as 128×128, also used grayscale image with 1, so the input to the network is in the form of $16 \times 128 \times 128 \times 1$ [92]. The encoder is used to extract spatio-temporal features from the input video which is a four-layer 3D convolutional network [92]. The various layers of 3D convolutional encoder consist of batch normalization (helps in accelerating the convergence at the training state), Leaky ReLU (used after batch normalization helps in giving a small positive value when the model is not active), 3D-pooling [92]. The final hidden layer encoder has the shape of $2 \times 16 \times 16 \times 64$, which has spatio-temporal information from the video clips provided as the input [92]. The decoder part of the model consists of reconstruction branch (stack of three 3D convolutional layers, helps in reconstructing the input from the hidden layer of the encoder) and a prediction branch, without 3D-pooling layer as in the encoder. The last layer of the decoder contains a sigmoid function, that is responsible to normalize the input data [92]. The reconstruction branch is used to reconstruct the past sequence from the output of the encoder. Then future frames are predicted using the prediction branch of the decoder part.

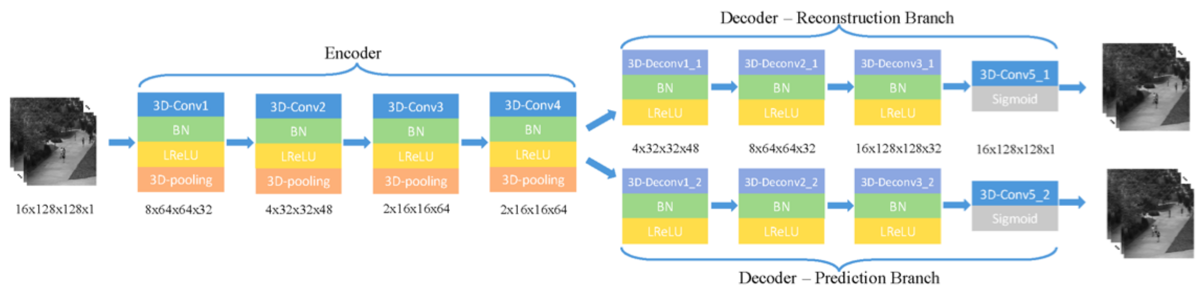


Figure 3.2: Architecture of 3D convolutional Autoencoder [92]

Prediction loss for the model is calculated using the following equation,

$$L_{pred} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T^2}\sum(T-t) \parallel X^t_{i+T} - f_{pred}(X_i)^t \parallel_2^2 \; [92] \tag{3.2}$$

Where X_i, X_{i+T}, t, $f_{pred}(X_i)$ denotes the input, ground truth of predicted frames, frame number, output of prediction branch respectively [92]. The method is evaluated using UCSD pedestrian, CUHK Avenue and traffic datasets. All these datasets are video anomaly detection datasets that contain video clips along with the anomalies in various real-time scenarios. The metrics used for evaluation include Receiver Operating Characteristic (ROC) curve, Area under the curve (AUC). The evaluation depicts that the future frames are predicted by the model to some extent, but it fails to predict anomalous event such as entering of the new vehicle into the scene [92]. Also, the model cannot be applied to complex and real-time scenarios [92].

3.1.2 Recurrent Models

Recurrent models can be used for video sequences predictions, means for analysing spatio-temporal representations of data provided in sequential form. These recurrent models can be used for long term predictions [76].

For anomaly detection one of the approaches uses a video prediction network constructed using U-Net architecture [47]. Figure 3.3 shows a basic example of the anomaly detection from the predicted video frames along with the difficulty in predicting abnormal events. The figure depicts the input sequence of images given by $I_1, I_2, ...I_t$, the predicted frame denoted by $\hat{I}_{t+1}$, and ground truth values denoted by I_{t+1} in normal and abnormal event scenarios [47]. It is noted that the predictions for the normal events are clear and correct, whereas the prediction of abnormal events is blurry [47]. Blurry predictions are produced in case of abnormal events due to the presence of higher reconstruction errors. To reduce these errors, the proposed approach creates a new video prediction method along with anomaly detection. The video prediction model is provided with a video clip, the objective of the model is to anticipate the future frames based on these past frames provided [47]. The model training is performed only using nominal data [47]. This approach is suitable for anomaly detection in predicted frames and also more realistic frames are predicted using this approach.

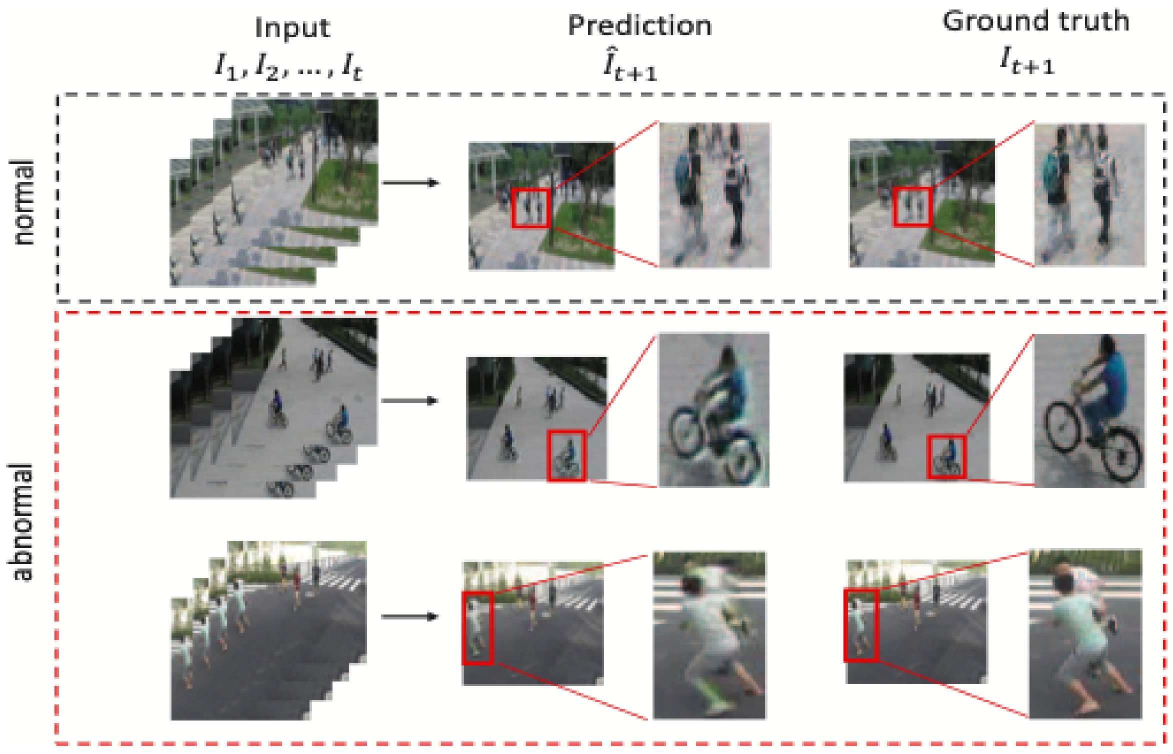


Figure 3.3: Video prediction along with anomaly detection [47, p. 6536]

Figure 3.4 shows the video prediction model used for anomaly detection. In this architecture, the U-Net predictor is considered as the generator module of GAN architecture [47]. Initially, the predictor is provided with image sequences, which predicts the future frames. The Flownet module is used to obtain the optical flow information, which is pretrained based on the previous frames [47]. To avoid blurry predictions and to produce higher quality images, intensity loss, gradient loss and optical flow loss constraints are used [47]. Finally, the discriminator module is used to identify whether the predicted frame is original or fake [47]. Intensity loss and gradient loss are used to produce the predictions that are closer to the ground truth values [47]. Sharpen images are formed by reducing the distance between the predicted frames and ground truth values. The method is evaluated suing CUHK Avenue, UCSD, ShaghaiTech dataset and showed that the method is able to detect and classify nominal and anomalous frames. Receiver Operation Characteristics (ROC) curve is initially formed to evaluate the performance of the anomaly detection model [47]. Then the area under the ROC curve (AUC) provides an accurate evaluation of model performance and higher values of the metrics show better performance of the model [45].

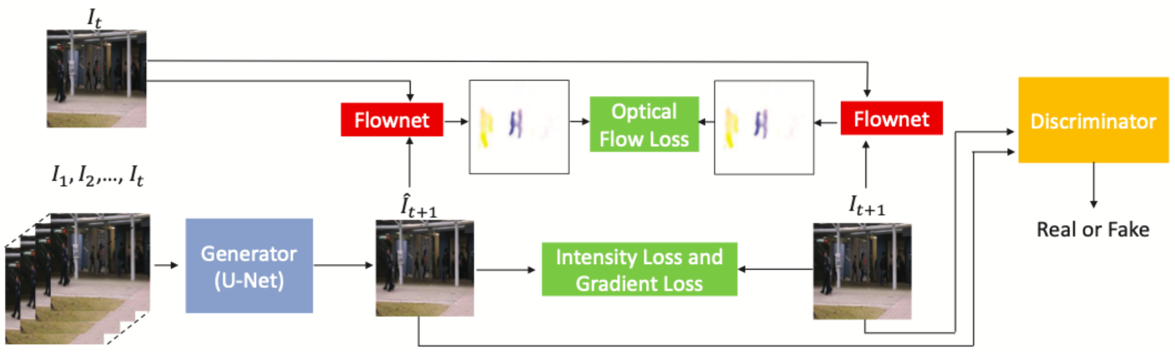


Figure 3.4: Video frame prediction network with predictor using U-Net architecture [47, p. 6537]

For long term predictions, the main idea is to identify those factors that vary over time. One of the approaches uses LSTM along with analogy-based encoder-decoder CNNs[76] for long term predictions and also to reduce blurry predictions. Blurry predictions are avoided by considering high-level features [76]. This is a hierarchical method, in which ground truth or higher-level structures are required during training time, this limits the application of this method [76]. In this method, video frames are predicted independently using the encoder-decoder architecture. The approach presented here is that the algorithm estimates the human pose given by XY coordinates which are then used for estimating the structure [76]. This structure is then used for predicting the future sequences in a cyclic manner [76]. In the last step, the algorithm observes the last observed frame, try to evaluate its structure and then generate future frames [76]. The basic steps involved are posed estimation, pose prediction and finally image generation. In this approach, LSTMs observes the input data and provide a set of XY coordinates to image generator [76]. The approach is evaluated using Penn Action dataset, that contains various video sequences of different actions [76].

Another approach uses the encoder-decoder network presented in [83] for hierarchical long-term video prediction. The main difference from the above method is the usage of low-level features for high-level features prediction and vice-versa [83]. The model architecture is similar to the above mentioned method except there is no requirement of landmark annotations in this approach [83]. The hierarchical prediction model is formed as follows: initially, image is predicted using high-level features obtained from context frames [83]. The future landmarks that are obtained from the context frames are predicted using LSTM [83]. Video frames are obtained from the predicted frames [83]. Once the above-predicted values are obtained, Visual Analogy Network (VAN) produces the image at a time t [83]. The VAN then recognize the high-level features and the probabilities, which is then converted into heat maps [83]. Finally, the network recognizes the difference to the image at time C to obtain the next predicted frame [83]. The predictor network of this approach contains LSTM and VAN [83]. The equation for predictor LSTM is given as,

$$\begin{cases} [\hat{e}_t, H_t] = LSTM(e_{t-1}, H_{t-1}) if, t \leq C \\ [\hat{e}_t, H_t] = LSTM(\hat{e}_{t-1}, H_{t-1}) if, t > C \end{cases} \quad [83] \tag{3.3}$$

Where $\hat{e}_t$, $\hat{e}_{t-1}$ denotes feature vector predicted by LSTM, feature vector obtained from the input image using encoder-decoder network respectively [83]. In this approach, the first frames of the input video are used to calculate future frames using the visual analogy method [83]. The method used to combine predictor LSTM and VAN is end-to-end (E2E) connection [83]. The usage of L2 loss directly can cause blurry prediction as a problem in [76]. For reducing this, the approach uses a feature encoder to train the LSTM network. The evaluation is performed using Human 3.6M dataset and the prediction is obtained for about 20 seconds [83]. The approach presented in this paper is suitable for long term predictions without the need for ground-truth high-level feature values [83].

3.1.3 Generative Models

The generative models are the most common approach used for video prediction tasks because these models generate data similar to the input, make this suitable for video prediction. A type of generative models is GAN, consists of a generator and discriminator to produce realistic frames. Generator (G) module is responsible to create new samples, whereas the actual image is separated from the generated images by discriminator (D) module [44]. The issue when blurry predictions are produced by MSE, L_2 losses can be solved by generative models from which sharper predictions are produced [44].

Blurry predictions that are caused by accessing the pixel values directly is a major problem in video prediction [44]. One of the approaches using dual GAN reduces the chances of blurry predictions using the optimized future frame generator and future flow generator [44]. The dual GAN architecture used in this paper as shown in figure 3.5 has three fully-differential modules such as probabilistic motion encoder (detects the motion and produces the motion frames from previous frames and provides as input to the generator), future frame generator (predict future frames by frame discriminator or flow discriminator) and future-flow generator (future flow is predicted using flow or frame discriminator) [44]. The dual GAN architecture contains two generators and two discriminators for achieving sharper and realistic frame predictions [44]. For the dual motion, the video sequence is fed as input, predicting the next frame by combining future frame prediction and future flow generator [44]. In this method, future flow is taken as two tasks(future frame prediction and then analysis using frame discriminator), and also probabilistic motion encoders are used for smooth flow prediction [44]. Probabilistic motion encoder firstly maps video sequence into code for obtaining latent representation (formed from convolutional network layers), which is further decoded using the decoder to predict future frames and flows [44]. A motion discriminator classifies the real and synthetic frames and the flow estimator estimates the flow using the predicted frame and real frame [44]. The dataset used for training is KITTI dataset and for testing Caltech Pedestrian dataset [44]. For evaluating predicted video quality, MSE, PSNR, SSIM metrics were used. Better performance is achieved using this learning method because it combines frame and flow-based mechanisms using dual GAN models.

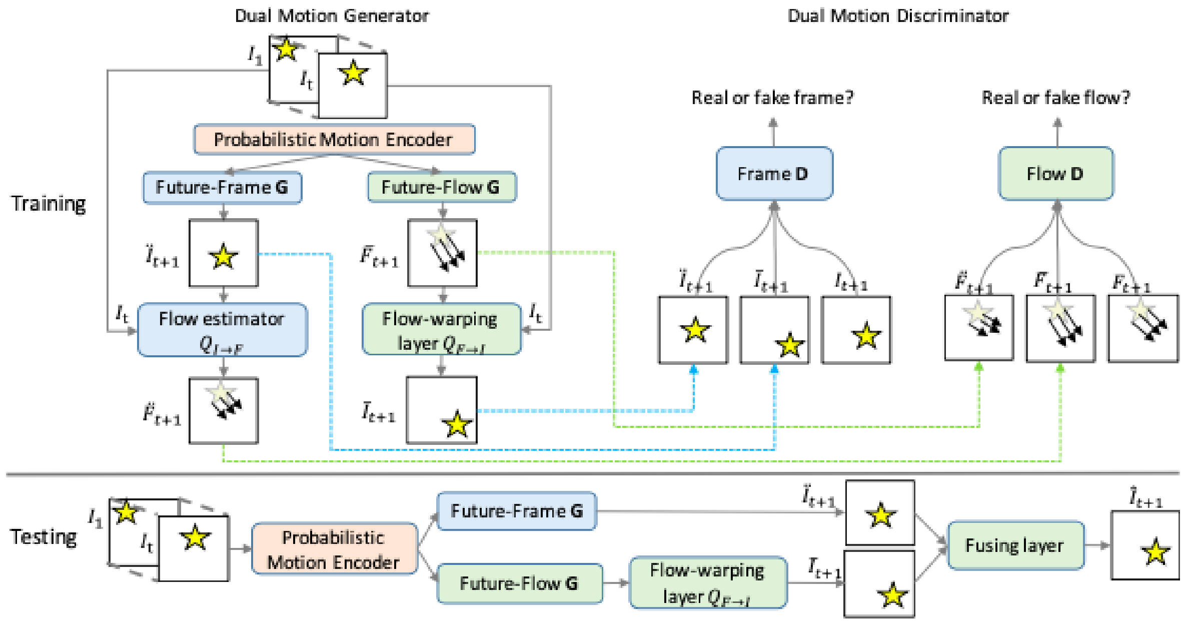
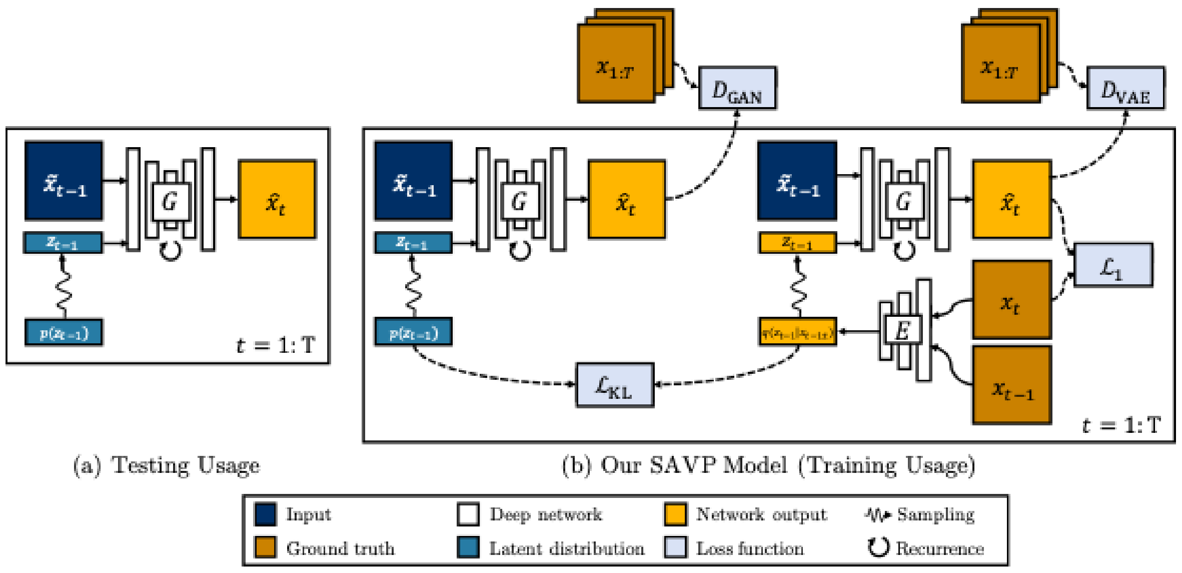


Figure 3.5: Dual GAN architecture for video prediction [44, p. 1746]

To avoid blurry predictions and to produce realistic predictions, one of the approaches uses a combined Variational Auto Encoder (VAE)-GAN architecture as the Stochastic Adversarial Video Prediction (SAVP) model [41]. The usage of VAE alone can cause blurry or unrealistic predictions due to presence of MSE in the posterior Gaussian distribution [41]. The usage of GAN networks solve the issues with blurry predictions, but still have issues with collapsing the model. To avoid these issues the approach uses the combined VAE-GAN [41]. Figure 3.6 shows the Stochastic Adversarial Video Prediction (SAVP) model with testing and training phase. The SAVP model contains a recurrent generator network denoted by G, acts as the video prediction model [41]. The model takes the input sequence x_o along with a sequence of latent codes $z_{0:T-1}$ to predict the future sequences of the image $\hat{x}_{1:T}$ [41]. The future frames are predicted by the recurrent generator using the latent code and information from previous frames [41]. The previous frames can be previously predicted frames or ground truth frame [41]. In the training phase, the generator is provided with the frames obtained by sampling the latent code z from the prior distribution $p(z)$ along with the previous frames [41]. The distributions used for sampling latent codes are prior and posterior distribution. The obtained frames are again fed back to the generator during the next iteration. After these input are fed to the generator, the frames are predicted and is denoted as $\hat{x}_t$. [41] In the training phase, the generator is trained to predict frames that has similarity with the real videos, which is done using the learned discriminator [41]. The encoder E is used to obtain the information during the transition from $\hat{x}_t$ to $\hat{x}_{t+1}$ [41]. The latent code sampled from posterior distribution is used for training [41]. Latent code is represented as $q(z_t|x_{t:t+1})$ [41]. The model uses latent code during training phase because it has information about the reconstructed frame $\hat{x_{t+1}}$. The generator used in this approach is made up of convolutional LSTM, that is responsible for predicting the variation between the current and the next frames [41]. The approach is evaluated on BAIR robot pushing and KTH dataset and showed higher accuracy [41]. This method is suitable to produce realistic diverse predictions also avoids problems associated with blurry predictions [41].



(a) Testing Usage (b) Our SAVP Model (Training Usage)

Figure 3.6: Architecture of Stochastic Adversarial Video Prediction (SAVP) model with VAE-GAN [41, p .7]

3.2 Datasets

Human3.6M Dataset [33]

Human3.6M dataset includes 3.6M video frames having human poses and the various images corresponding to these poses [33]. In this dataset, the motions are performed by 11 actors in 17 scenarios such as discussion, smoking, taking photos, posing, sitting on a chair and so on. These data are collected using four digital cameras [33].

KTH dataset [70]

KTH dataset is a human action recognition dataset that contains video database with six different human actions such as jogging, running, walking, boxing, hand-clapping and hand waving by 25 actors [70]. These actions are performed in 4 different scenarios such as outdoors (s1), outdoors with scale variation (s2), outdoors with different clothes (s3) and indoors (s4) [70]. This dataset contains 2391 sequences which are captured using a camera having a frame rate as 25 frames per second [70]. The original images are of size 120×160, which is resized to 64×80 to reduce the complexity [70]. The dataset contains 216 sequences of testing images and 383 sequences of training images [70]. This human action recognition dataset has been used for video prediction in [41].

CUHK dataset [49]

CUHK avenue dataset is used for detecting abnormal events in video clips and consists of 21 testing and 16 training video clips [92]. There is a total of 30652 frames in the video clips and the anomalies considered are loitering, throwing, or running [49].

KITTI dataset [25]

KITTI dataset is a vision-based dataset that contains 7481 training images. These data are captured from a moving platform and includes laser scans, camera images, GPS measurements. This dataset is used for video prediction tasks in [44]. The raw dataset is separated into various categories such as 'city', 'road', 'residential' and so on. The size of KITTI dataset is 180GB [25].

Robot Pushing dataset [17]

Robot pushing dataset is a action-conditioned video prediction dataset collected from 10 different robotic arms [17]. The dataset includes 1.5 million video frames having 57000 interaction sequences produced by moving different objects by the robotic arms [17]. This dataset includes one training set and two test set of previously seen and unseen objects [17].

UCSD Pedestrian dataset [8]

UCSD pedestrian is an anomaly detection dataset with videos of pedestrians on the walkways obtained using the fixed camera. The videos collected are in 8-bit grayscale having 238×158 as dimensions with 10 frames per second. This dataset has videos from two scenarios such as Ped1 and Ped2 [92]. A total of 70 video clips are available in Ped1 scene with 200 frames in each clip and 28 video clips in Ped2 scene with 170 frames in each clip [92]. The anomalies considered in the dataset are bicycle, wheelchairs, vehicles, skateboards [92].

Penn Action Dataset [76]

This dataset can be used for various human action prediction training sequences. This dataset consists of 2326 video sequences with 13 human joint annotation and 15 different human motions for each of sequence provided.

Moving MNIST dataset [16]

Moving MNIST dataset consists of 10000 sequences having a length of 20 seconds with 2 digits moving. The dataset contains randomly generated grayscale images of size 64×64 [16]. This dataset can be used for long term prediction. This dataset can be used for unsupervised learning techniques during the training phase.

3.3 Metrics

MSE [44][63][78][48]

Mean Squared Error (MSE) is calculated as " the average of the squared difference between expected value and actual value" [44]. The MSE is calculated as follows,

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2[44] \tag{3.4}$$

Where Y is the actual value and $\hat{Y}$ is the expected value.

PSNR [44][47][63]

The image quality can be assessed by using a popular approach called PSNR [44]. A higher value of PSNR indicates the better quality of image. The formula used for calculating PSNR is as follows,

$$PSNR(I, \hat{I}) = 10log_{10}\frac{[max_{\hat{I}}]^2}{\frac{1}{N}\sum_{i=0}^{N}(I_i - \hat{I}_i)^2}[47] \tag{3.5}$$

Where I is the ground truth, $\hat{I}$ is the predicted frame, N is the total number of iterations [47].

SSIM [31][48]

The image quality can be measured by another popular approach called SSIM [31]. A higher value of SSIM indicates better predicted frame quality [31]. SSIM is calculated as,

$$SSIM(f, g) = l(f, g)c(f, g)s(f, g)[48] \tag{3.6}$$

Where f, g, l, c, s represents reference image, text image, luminance comparison function, contrast comparison function, structure comparison function respectively [31]. The other metrics used for evaluating the predicted frames are **Learned Perceptual Image Patch Similarity (LPIPS)** [7], **Mean Absolute Error (MAE)** [32], **L1** loss [63]

3.4 Summary

In the case of video prediction, the input to the prediction model is a past video frame or image and the output is the future predicted frame. For the video prediction tasks, generative models, recurrent models and feedforward architectures can be used. Most of the approaches use recurrent models for long-term predictions, but the usage of recurrent models alone can cause blurry predictions. This is because the output from the one state is fed as input to the next state, so if there is some error in that output, that error will be propagating. These blurry predictions caused by recurrent models are avoided using various generative models. Feedforward architectures can be used for more realistic and high-quality frame prediction. Some of the approaches use feedforward architecture for video prediction and also then detecting anomalies in the predicted future frames. Based on the previous explanations, it is difficult to identify the most common approach for video prediction, because video prediction methods are application-specific. There are existing challenges in video prediction such as early frame prediction, complex scenarios.

The datasets used are common for most of the approaches. One of the famous video prediction datasets is robot pushing dataset that contains various actions performed by a robotic arm. Other famous datasets include KTH, CUHK, UCSD Pedestrian, KITTI, Penn Action dataset. All these datasets contain various actions performed by humans in the form of video clips. One of the action anticipation datasets, UCF101 dataset is also used for video prediction applications.

The evaluation metrics used for video prediction measure the image quality of the predicted frame. The metrics include MSE, PSNR, SSIM. The blurry predictions are due to the presence of MSE losses. These metrics are also not sufficient for evaluating the performance. All these findings are tabulated in Table 3.1.

Reference	Method used	Dataset	Contribution	Drawback
Feedforward architectures				
Zhao et al.[92]	STAE	UCSD Pedestrian, CUHK Avenue, Traffic dataset	Video prediction and anomaly detection	Fails for some real-world scenarios
Gao et al.[22]	DPG	KITTI, CalTech pedestrian	Combined flow predictor and occlusion inpainter for video prediction	-
Reda et al.[63]	3D CNN and Spatially-displaced convolution(SDC)	CalTech pedestrian, YouTube-8M	High-resolution frame prediction from past frames	Not suitable for multi-scale architectures
Vukotič et al.[78]	Encoder-decoder neural network	KTH	One-step long-term video prediction	Low resolution predictions
Lotter et al.[48]	Predictive neural network(PredNet)	KITTI, CalTech Pedestrian	Predictive coding for video prediction	-
Recurrent models				
Villegas et al.[76]	LSTM+ analogy-based encoder-decoder CNN	Penn Action dataset, Human 3.6M	Long term prediction	Less accuracy of prediction(generates only single trajectory)

Table 3.1: Summary of methods, datasets used for video prediction

Reference	Method used	Dataset	Contribution	Drawback
Wichers et al.[83]	LSTM and VAN	Human 3.6M	Hierarchical long term prediction	Predictions disappear occasionally
Castrejon et al.[7]	Hierarchical Conditional VariationalRNN	Moving MNIST, BAIR push	Video prediction with good likelihood	-
Medel et al.[52]	Convolutional LSTM	UCSD Pedestrian, Avenue, Subway Datasets	Method to predict future frames with anomaly detection	Fails to detect some anomalous events
Oliu et al.[56]	Folded RNN (fRNN)	Moving MNIST, UCF101, KTH	A shared recurrent model that needs less memory and less computational cost	Blurry predictions
Hosseini et al.[32]	Inception-inspired LSTM	KITTI, KTH	Self-supervised approach for video prediction with better accuracy	High computational cost
Wang et al.[82]	Eidetic 3D LSTM	Moving MNIST	Long-term video prediction using short-term motions as input	Requirement of labelled dataset for evaluation
Ye et al.[90]	Entity predictor-frame decoder	Penn Action dataset	Pixel-level future prediction	-
Finn et al.[17]	Convolutional LSTM	Robot pushing, Human 3.6M	Action-conditioned video prediction model	Application specific
Fan et al.[16]	Cubic LSTM	Moving MNIST, Robot pushing, KTH	Accurate video prediction approach in which spatio-temporal data is processed separately, reducing the difficulties in prediction	-
Generative models				
Liang et al.[44]	Dual GAN network	KITTI dataset, Caltech Pedestrian dataset	Future frame prediction using dual GAN mechanism	Not suitable for real-world and complex scenarios

Table 3.1: Summary of methods, datasets used for video prediction

Reference	Method used	Dataset	Contribution	Drawback
Kwon et al.[40]	Retrospective Cycle GAN	KITTI, UCF101, CUHK Avenue	Future frame prediction and anomaly detection using a generator and two discriminator framework	-
Li et al.[42]	Spatial-temporal conditional VAE	KTH	Multi-step prediction network with single still image as input	-
Lee et al.[41]	VAE-GAN	BAIR action-free robot-pushing dataset and KTH	Stochastic video prediction model	-
Liuwen et al.[47]	Video prediction network using U-Net	CUHK Avenue, UCSD, ShanghaiTech	Future frame prediction network is used for anomaly detection	Framework fails for some real-world applications

Table 3.1: Summary of methods, datasets used for video prediction

4

Trajectory Prediction

Trajectory prediction refers to predicting future trajectories by using historical trajectories as input. Trajectory prediction finds application in fields of autonomous driving, transportation systems, mobile communication systems [79]. Trajectory prediction can be a future pedestrian trajectory prediction [55] or vehicle trajectory prediction [3]. Pedestrian trajectory prediction refers to predicting future positions of a pedestrian in terms of location coordinates based on historical pedestrian positions [88]. Trajectories are defined by a set of positions and the change in position with respect to time [55]. This can be called as finding acceleration or velocity components associated with vehicles. In the case of the autonomous system, the forecasting about the trajectory has many applications such as decision making, motion planning. Figure 4.1 shows a basic model of trajectory prediction.

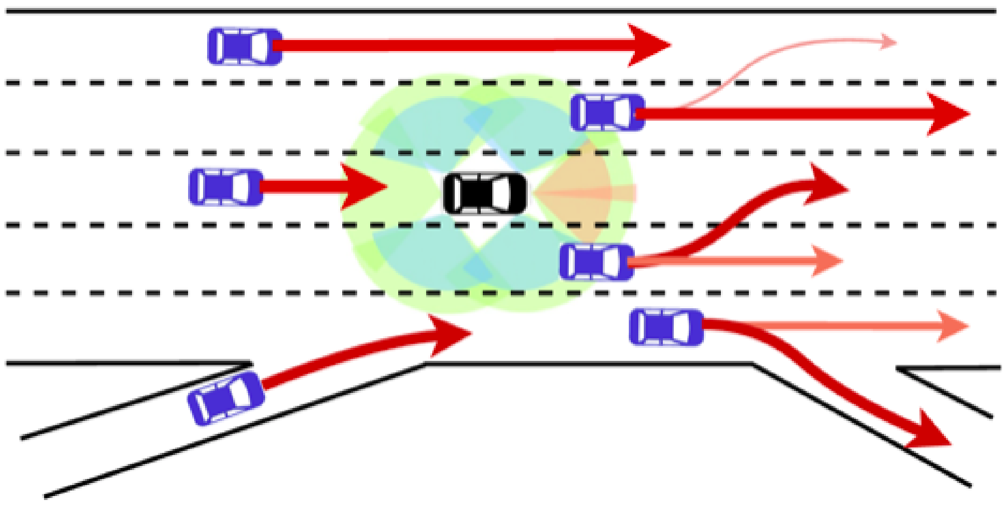


Figure 4.1: Trajectory prediction in case of autonomous vehicle [14, p. 1581]

In the figure, the black car is the autonomous car, whose trajectory needs to predicted with respect to the surrounding vehicles. As observed from the figure, knowing trajectory in advance is better way to avoid collision in the case of these self-driving systems. Various approaches can be used for finding the future trajectory based on the past observed trajectory data. In this section, various trajectory prediction methods, datasets used, and metrics used for evaluation will be discussed.

4.1 Methods used

4.1.1 Feedforward Architectures

One of the main problem associated with RNN is the delay in long-term predictions, caused due to the recurrent nature of operation of the model. One of the approaches presented in [55] uses CNN for trajectory prediction, which solves the problem associated with RNN. The basic CNN architecture is a connected network, for which the input is the historical trajectories and is passed through these connected layers [55] as shown in Figure 4.2. To maintain temporal dependency, the various layers of CNN is stacked [55]. The final predictions are made by joining all the features obtained from the previous connected layer. The final predictions are denoted by $(x^t, y^t)_{t=t+1}^{t+t_{pred}}$ [55].

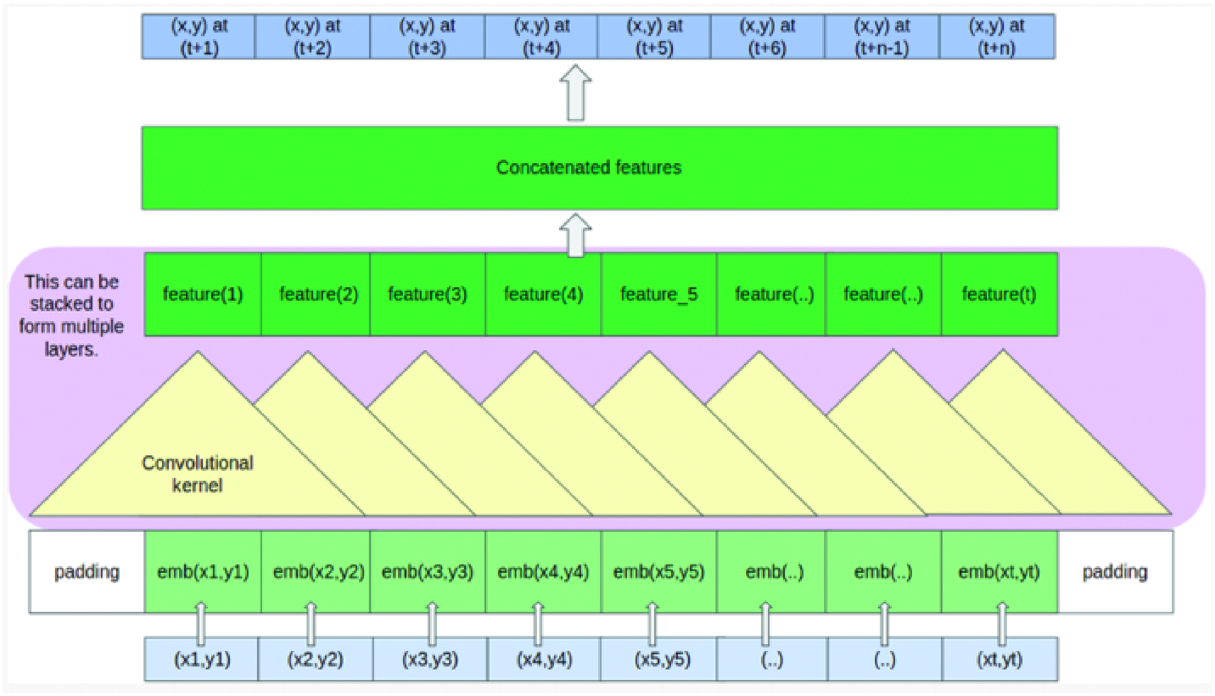


Figure 4.2: CNN architecture for predicting trajectory [55, p. 4]

The continuously predicted pedestrian trajectory is the advantage of this approach. In this approach, the input is constantly given to the convolutional layer to maintain same number of layers in the input and output. For the human-trajectory prediction evaluation, ETH and UCY datasets are used. These dataset contains various human-interaction scenarios including collision avoidance, dispersal, people crossing various paths, trajectories. The metrics used for finding prediction error are Average Displacement Error

(ADE) and Final Displacement Error (FDE) [55]. The real-time trajectory predictions for the autonomous systems can be achieved by using these CNN approach [55].

4.1.2 Recurrent Models

For trajectory prediction applications, RNN approaches are better, because they consider previous positions of the trajectory, which is important in trajectory prediction. The problem associated with RNN is the delay in long-term predictions.

For predicting trajectories of vehicles in highway, one of the approaches presented in [3] uses LSTM network. The sudden change in velocity and lane is a major challenge in predicting trajectory in highway [3]. Next Generation Simulation (NGSIM) which is largest dataset containing trajectories of vehicles is used [3]. The dataset provides trajectories as (X, Y) coordinates of the vehicle's front in the global frame and (x, y) coordinates provides the value associated with the alignment with the road(local frame) [3]. The value of x indicates the lateral position, whereas y indicates the longitudinal position of the vehicle [3]. The limitation of this dataset is the presence of noise due to trajectories are collected as videos, which is avoided using the filter in this approach [3]. In this approach, the vehicle trajectories are predicted based on the coordinates of vehicles surrounding them [3] as shown in Figure 4.3.

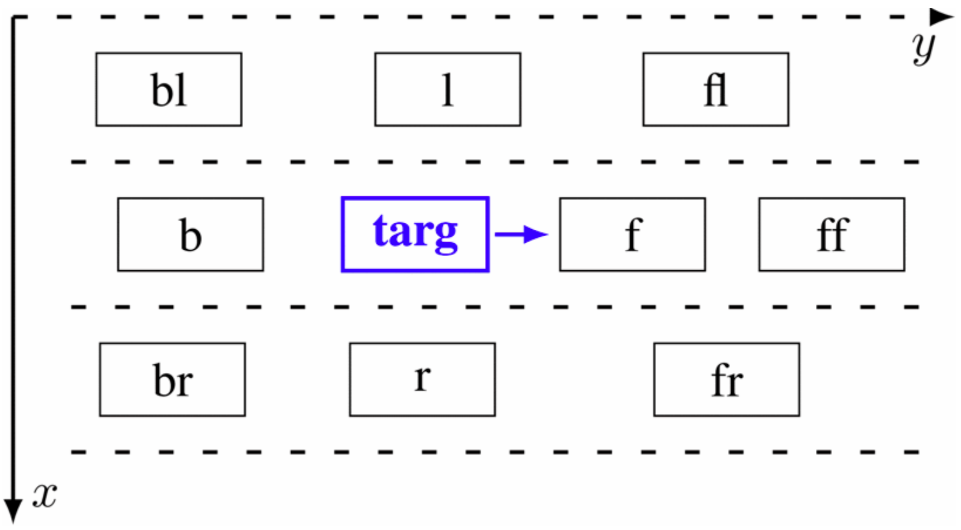


Figure 4.3: Target vehicle in local axis system along with the vehicles surrounding it [3, p. 355]

The various features considered are local longitudinal position (y_{targ}), local lateral position(x_{targ}), corresponding velocities and also various features related to surrounding vehicles [3]. The objective of this approach is future trajectory prediction of the target vehicle [3]. It is defined with higher longitudinal values [3]. So instead of predicting longitudinal positions, the velocities of the target vehicle is anticipated and is denoted as $\hat{v}_y targ$ [3]. The lateral values are predicted as it is and denoted as $\hat{x}_{targ}$ [3]. The final predicted trajectory is of the form $[\hat{x}_{targ}, \hat{v}_y targ^k]$, where the value of 'k' represents the time [3].

In trajectory predictions, the goal is predicting the next positions (x, y) of the target vehicle, and it is considered as a regression problem [3]. Figure 4.4 shows the internal structure of the LSTM used for this approach. In the figure, m_t denotes the cell's internal memory, h_{t-1} denotes previously obtained output, x_t denotes the new input and m_{t-1} denotes the previous state [3]. The different operations performed by LSTM cells are called as gates. The forget gate receives the previous input and passed to output gate along with input gate [3]. The input gate is provided with that data from h_{t-1} and x_t and determines the memory needed for the storage based on the inputs. This structure of LSTM helps in providing long-term predictions. In this application, 256 LSTM layers are stacked together, which helps in getting features from the input, which is then passed over the dense layer for obtaining the final predicted trajectory [3]. The model is evaluated using Root Mean Square Error (RMSE) calculated between the prediction network and the expected value [3]. The usage of LSTM cause delays in prediction, which is a deficit of this approach for long-term and accurate highway trajectory prediction [3].

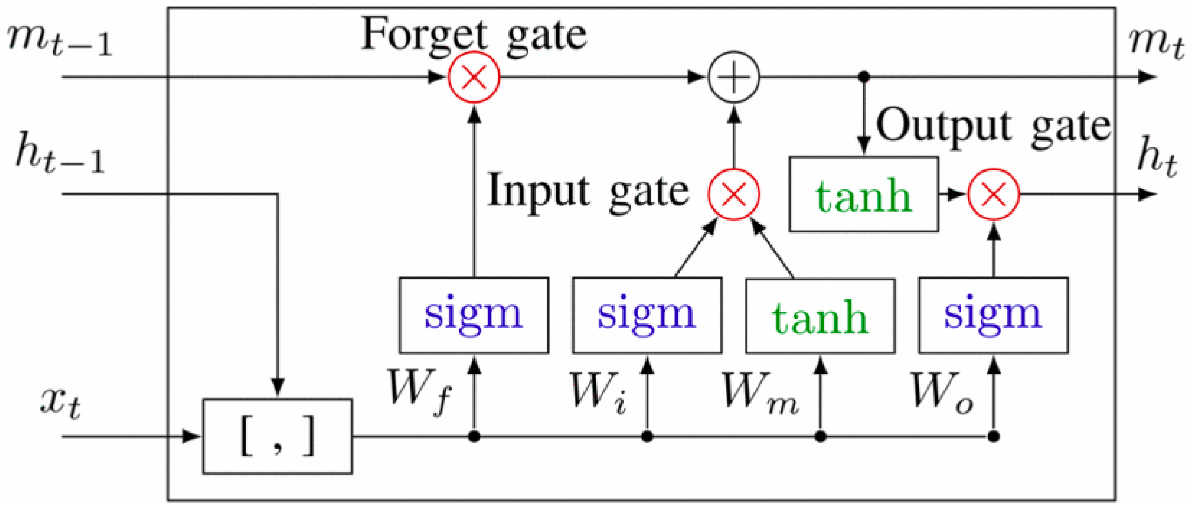


Figure 4.4: LSTM internal structure for trajectory prediction [3, p. 356]

For the pedestrian trajectory prediction, an LSTM based approach presented in [88] can be used. The architecture used in this approach is known as Crowd Interaction Deep Neural Network (CIDNN) [88] as depicted in Figure 4.5. The future pedestrian locations are predicted by finding the displacement coordinates using this approach [88]. The final pedestrian position is calculated in terms of speed and acceleration. To obtain this motion-related information, LSTMs are used, with the input as the location coordinates of pedestrians. The coordinates of the pedestrian location are provided to the multi-layer perceptron to find the relation between the pedestrian position and the predicted target location [88]. In this approach the model approximate the coordinates of the location at every time step as sequential, by which pedestrian location changes are predicted [88]. As shown in the figure, the architecture of CIDNN consists of four modules such as location encoder, displacement prediction, crowd interaction and motion prediction [88]. Motion encoder module is used to obtain the motion coordinates of pedestrian including velocity, acceleration using LSTM [88]. In this approach, two LSTMs are stacked together for obtaining motion information [88]. The historical trajectory from every pedestrian is provided as the input to these stacked LSTM. The output (z_t^i) from these stacked LSTMs are given as,

$$z_t^i = f(S_1^i, ..., S_t^i)[88] \tag{4.1}$$

Where S_t^i denotes the spatial location coordinates of the pedestrian [88].

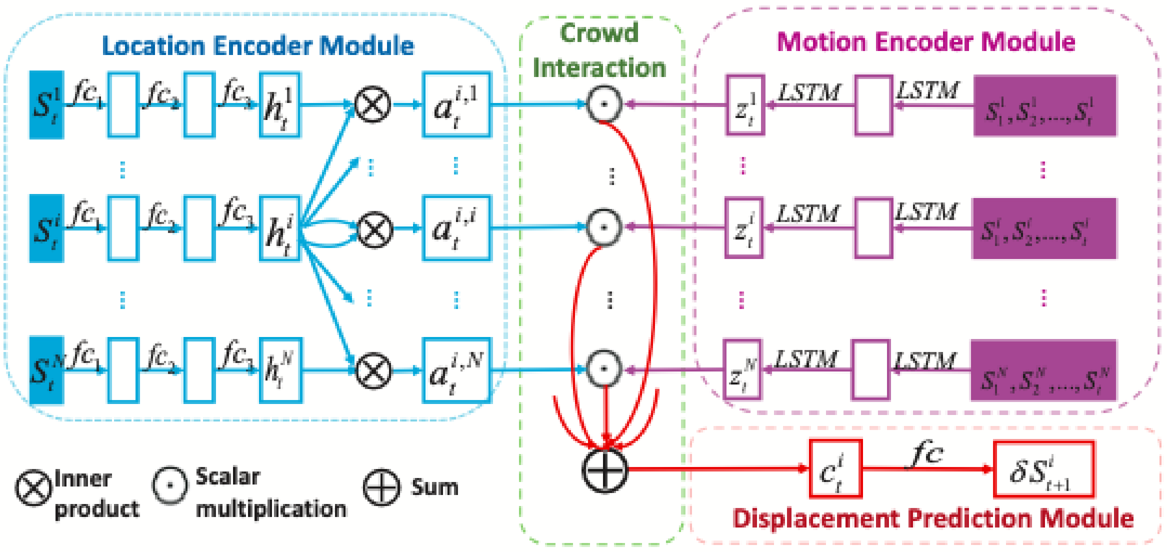


Figure 4.5: Crowd interaction deep neural network (CIDNN) architecture for pedestrian trajectory prediction [88, p. 5275]

The hidden layers of the LSTM nodes are fixed to 100 in this approach [88]. The location encoder module is made up of multi-perceptron having 3 layers with ReLU as the activation function [88]. The equation to calculate output of location encoder (h_t^i) is given as,

$$h_t^i = g(S_t^i)[88] \tag{4.2}$$

The output from the location encoder module is given as the input to the crowd interaction module, by which the spatial relation between the pedestrians can be calculated. The crowd interaction module does a scalar multiplication between location encoder and motion encoder output. The output from the crowd interaction module is calculated as,

$$c_t^i = \sum_j a_t^{i,j} z_t^j [88] \tag{4.3}$$

The final module is the displacement prediction module [88]. This module is a fully connected layer summing overall pedestrian location at a particular time, and final pedestrian position is predicted [88].

The final prediction of the displacement module is given as,

$$\delta s_{t+1}^{i} = W c_{t}^{i} + b[88] \tag{4.4}$$

Where, W and b are weights and bias respectively, which are considered to be parameters in calculating final prediction value [88]. From the location displacement (δs_{t+1}^{i}) value the final pedestrian position p_i at the time t is calculated as,

$$t+1 : S_{t+1}^{i} = \hat{S}_{t}^{i} + \delta s_{t+1}^{i}[88] \tag{4.5}$$

This approach, the pedestrian trajectory is predicted for each pedestrian separately. Due to the less number of hidden nodes present in this approach, this implementation is suitable for even small pedestrian trajectory prediction applications. The datasets used for evaluation include UCY, ETH, CUHK crowd datasets [88]. To performance of method can be evaluated using Average Displacement Error (ADE) [88]. This approach is suitable for long-term pedestrian trajectory prediction applications. The model is evaluated in various crowded scenes and showed better accuracy in every scenario with better predictions. The deficit of the approach is the requirement of larger time for execution. Otherwise, this method is the most suitable for long-term pedestrian trajectory prediction models compared to other recurrent model based approaches [88].

4.1.3 Gaussian Process Models

Gaussian Process (GP) models are those models in which the generated data is assumed to follow a Gaussian distribution. For long-term trajectory prediction, one of the approaches presented in [28] uses Gaussian Process Regression (GPR) model. The objective is future trajectory prediction by using GPR model with the vehicles historical trajectory data [28]. A GPR model (f) is explained in terms of mean $m(X)$ and covariance $k(X, X')$ function [28] and is formulated as,

$$f \sim gp(m(X), k(X, X'))[28] \tag{4.6}$$

The kernel function used in this approach is the squared exponential kernel, and this kernel determines the value of the covariance matrix [28]. Figure 4.6 shows the GPR prediction model for vehicle trajectory prediction with an offline and real-time module [28]. Initially, the vehicle coordinates obtained from the sensors of the vehicle is provided to coordinate transformation, which transforms the vehicle local coordinates to global coordinate frame for giving proper input to the system, after which clusters are formed using k-means clustering using first and last coordinates of the data [28]. Next step is to learn the GP model so that predictions can be made. The motion model is given as the transformation of the vehicle coordinates (x,y) to their derivatives with respect to time and is denoted as v_x, v_y [28]. The trajectory prediction aims to predict next position, that is, if the model is provided with positions x^t, y^t, then the model should predict next positions x^{t+1}, y^{t+1} [28]. In this approach, the motion model is defined in terms of GPR, given as,

$$v_x \sim gp_x(m(p), k(p, p'))[28] \tag{4.7}$$

$$v_y \sim gp_y(m(p), k(p, p')[28] \tag{4.8}$$

Where p denotes the coordinates of the vehicle [28].

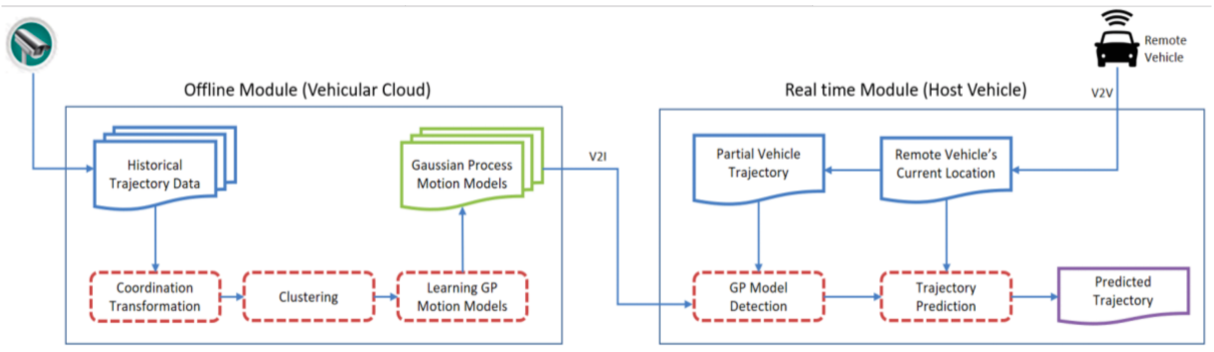

Remote Vehicle
Offline Module (Vehicular Cloud)
Real time Module (Host Vehicle)
V2V
Historical Trajectory Data
Gaussian Process Motion Models
V2I
Partial Vehicle Trajectory
Remote Vehicle's Current Location
Coordination Transformation
Clustering
Learning GP Motion Models
GP Model Detection
Trajectory Prediction
Predicted Trajectory

Figure 4.6: Trajectory prediction using GPR model [28, p. 552]

The model is then trained using the Lankershim Boulevard real-time trajectory dataset [28].From the above learned motion models, the best one is selected depending on the likelihood between the last location of the trajectory and the models [28]. After the correct model is obtained, the Gaussian distribution for all position is formed to predict the final trajectory [28]. The accuracy of prediction is evaluated using RMSE and showed higher accuracy for real-world scenarios [28]. The paper does not completely explain whether the approach correctly predicts and thereby avoid the collision.

4.1.4 Combined Models

For motion trajectory prediction, one of the approaches presented in [85] uses a sequential model with combined CNN-LSTM network for more accurate and faster predictions. Sequential model is the model in which the convolutional layers are stacked in a linear fashion. In this approach, CNN collects the data from the input and the LSTM network then predicts the trajectory. Figure 4.7 shows the internal structure of CNN-LSTM architecture presented in this paper. The first step is to identify the abnormal events and eliminating those values to get valid data, which is denoted by $'M'$ in the figure. This valid trajectory information is then normalized and separated into training and test data and is denoted as M_{train}, M_{test} respectively. Then the convolutional layer is provided with M_{train} values and is denoted as $M_c = M_{train} * K$, where $'K'$ is the kernel size [85]. Then the same training values are then passed to the pooling layer and is calculated as $M_p = max(M_c^{m,n})$ [85], where (m, n) is the size of pooling window. After passing the trajectory information through these convolutional layers, the features of the input data is obtained, which is then passed through the LSTM for predicting the future trajectories [85]. The final result of passing M_p to LSTM is denoted as $M_L = M_p \to LSTM(f_t, i_t, c_t, o_t)$, where f_t, i_t, c_t, o_t denotes various gates of LSTM [85]. Finally, the prediction model is formed by training the model as shown in Figure 4.7. The model is trained using NGSIM dataset, which is preprocessed to remove outliers thereby increase the accuracy of prediction [85]. In this case, to remove outliers in the data, the box plot is considered, because the data being processed does not need to be any specific distribution.

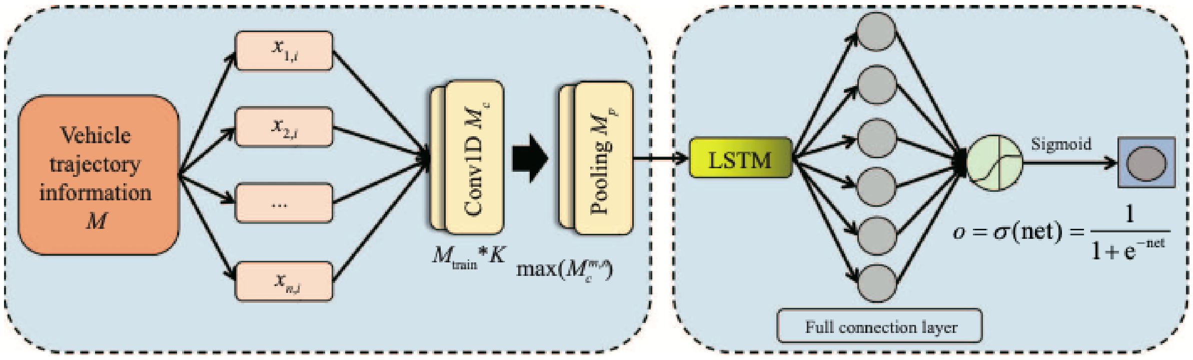


Figure 4.7: CNN-LSTM internal architecture used for trajectory prediction[85, p. 4]

CNN is a neural network architecture, that contains an input layer, followed by a hidden layer and finally output layers [85]. The hidden layer consists of pooling layer, fully connected layer and convolutional layer [85]. The parameters from the input data are extracted using the pooling layer and convolutional layer, these layers decrease the parameters required by the model, thereby increasing the speed of prediction. The convolutional layer is formed by the formula given below,

$$M_{c,t} = f(M_{train,t} * K + b) = f(\sum_j M_{train}, j * K_{t-j} + b_j)[85] \tag{4.9}$$

Where $M_{c,t}$ denotes final convolutional output, $M_{train,t}$ denotes data required at time t. The layer after the convolutional layer is known as the pooling layer, which is used to decrease the data size for easy processing [85]. The formula for obtaining pooling layer is given as,

$$M_p = max(M_{c,t}^{m,n})[85] \tag{4.10}$$

The features obtained from this pooling layer is then provided to LSTM network, which allows long-term predictions [85]. The LSTM model presented here contains an input gate, forget gate, hidden gate and output gate [85]. The working of this LSTM model is similar to the one explained in [3]. The steps involved in trajectory prediction are data preprocessing (abnormal values are eliminated and the dataset is split into train, test sets and provided as input to the CNN-LSTM architecture), sequential model (data after pre processing is provided to architecture), prediction model [85]. Finally, the model is evaluated using RMSE and MAE [85]. The value of RMSE near to zero indicates better performance of the model evaluated. The overall architecture of CNN-LSTM model is depicted in Figure 4.8. CNN model is used to extract features and LSTM model to predict the trajectory. This combined model provides faster prediction, thus it can be applied for real-world complex scenarios.

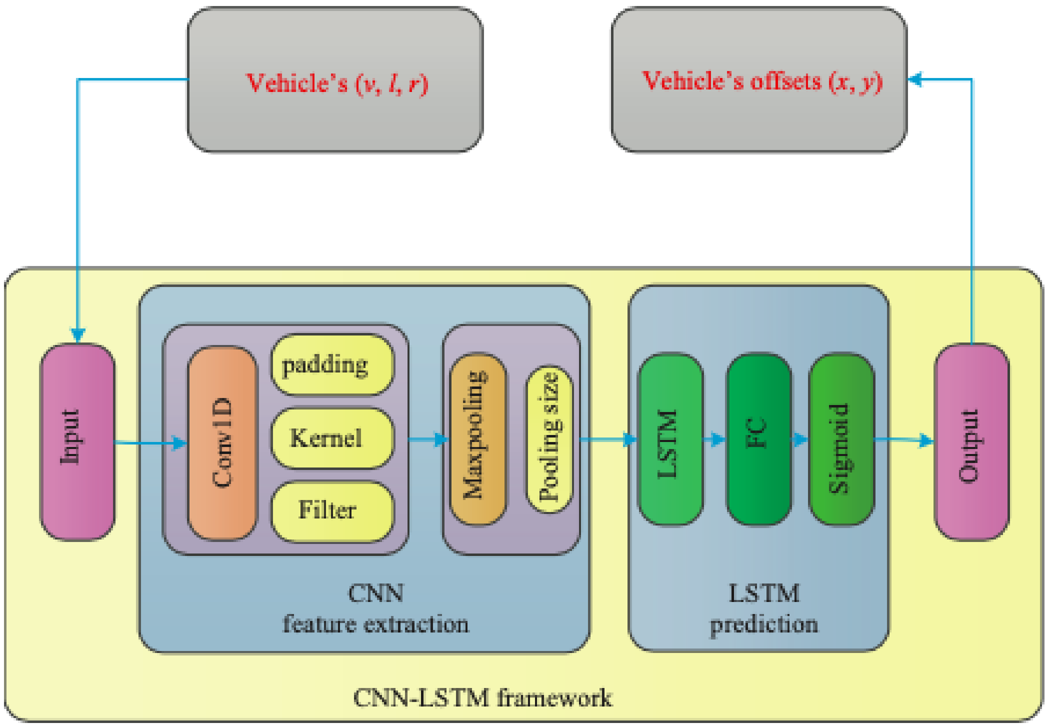


Figure 4.8: Overall CNN-LSTM architecture for trajectory prediction [85, p. 10]

4.2 Datasets

4.2.1 Vehicle Trajectory Datasets

NGSIM US 101 [11][14], NGSIM I-80, Lankershim Boulevard [28]

NGSIM is the trajectory dataset contains various vehicle trajectories in the form of videos obtained using the cameras [11]. The NGSIM dataset is divided into various sets based on the location from which the data is collected such as US 101, Lankershim Boulevard, I-80, Peachtree Street metadata [11]. The trajectory data contains exact location of the vehicle at every one-tenth of second, gives complete position of the vehicle relative to other vehicles in the lane [3]. US 101 dataset contains trajectory data of freeway, which contains details about lane changing in freeways. Lankershim Boulevard dataset contains 30 minute videos of vehicle motion collected between various intervals of times and can be used for various trajectory prediction applications [11].

TITAN [50]

The [50] uses a new TITAN dataset (created for this prediction) which consists of labeled video clips with videos about the moving vehicles in a highly diverse environment. The dataset consists of various labels such as actions, pedestrian action attributes, pedestrian age groups, vehicle states, and so on.

4.2.2 Pedestrian Trajectory Datasets

ETH [43]

ETH is a pedestrian dataset that consists of two scenes having 750 pedestrians in total [43]. The dataset is divided into two subsets as ETH and hotel. This dataset is used for pedestrian trajectory prediction applications [55][89].

UCY [43]

UCY is another popular pedestrian trajectory dataset, that contains video data obtained from 786 pedestrians with three subsets of data [43]. The subsets are named as ZARA-01, ZARA-02 and UCY [43]. This dataset is also used for pedestrian trajectory prediction in [89][55].

Town centre dataset [65]

Town centre dataset is a pedestrian trajectory dataset captured using camera in the crowded scenes. The dataset includes videos of 5 minutes duration with 230 hand-labeled video tracks [65]. This dataset is used for predicting pedestrian trajectory [89].

4.3 Metrics

ADE [55]

ADE is calculated as the average difference between distances of predicted trajectory with respect to the ground truth values [55] as given in the following equation,

$$ADE = \frac{\sum_{t=obs+1}^{T_{pred}} \parallel Y_t - \hat{Y}_t \parallel}{T_{pred} - T_{obs}} [55] \tag{4.11}$$

FDE [51]

FDE is calculated as the difference between true location and predicted final location [51] and is given as,

$$FDE = \frac{\sum_{i=1}^{N} \parallel \hat{X}_i^T - O_i^T \parallel^2}{N} [51] \tag{4.12}$$

Where $\hat{X}_i^T$ is the predicted trajectory at the time t and O_i^T is the original location [51]. ADE and FDE are the most common metrics used for evaluation of trajectory prediction.

RMSE [50][85][13]

RMSE is one of the main evaluation metric used for trajectory prediction. The values of RMSE near to zero indicates good prediction result showing the better performance of the model. The formula used for calculation of RMSE is given as,

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i^o - x_i^p)^2[85]} \tag{4.13}$$

Where x_i^o denotes ground truth data, x_i^p denotes predicted data.

4.4 Summary

For the trajectory prediction model, the input is set of trajectories denoted by the position of the vehicle with respect to various coordinates and the expected output is predicted trajectory, which can be predicted human or vehicle trajectory. Recurrent models networks are more suitable for human trajectory prediction or vehicle trajectory, but their delayed prediction limits its application. Recurrent models can be used for long-term predictions and complex scenarios. LSTM networks are used for trajectory prediction on highways and freeways. Feedforward architectures are not common for trajectory predictions, but they are suitable for immediate and realistic trajectory prediction applications. Gaussian models are the other most common approach for the long term and accurate predictions. Most of the applications use GPR models for predicting trajectories. The limitation of the recurrent model in this prediction can be avoided by CNN, combined CNN-LSTM, gaussian models.

The datasets are also common for most of the trajectory predictions. The vehicle trajectory predictions use NGSIM US-101, I-80, Lankershim Boulevard dataset that contains videos of vehicle positions in different scenarios captured using the camera. All of the pedestrian trajectory methods use UCY, ETH dataset that contains various actions performed by humans. There exist some limitations in the availability of datasets.

The most common metrics used for evaluation are ADE, FDE, RMSE. These metrics are not suitable for perfectly evaluating trajectory prediction algorithms. All these findings are tabulated in Table 4.1.

Reference	Type of prediction	Method used	Dataset	Contribution	Drawback
Feedforward architectures					
Yi et al.[91]	Pedestrian trajectory	Behavior CNN	Pedestrian Walking Route	CNN based approach to predict pedestrian behavior based on location coordinates	Not suitable for long-term pedestrian trajectory prediction

Table 4.1: Summary of methods, datasets used for trajectory prediction

Reference	Type of prediction	Method used	Dataset	Contribution	Drawback
Nikhil et al.[55]	Pedestrian trajectory	CNN	ETH, UCY dataset	Convolutional network architecture for immediate predictions	Fails for some real-time applications
Malla et al.[50]	Vehicle trajectory	Encoder-decoder Trajectory Inference using Targeted Action priors Network (TITAN)	TITAN	Predicting future trajectories of the agents of the scene from an egocentric point of view gained from a moving platform	Model is unable to predict the human interactions by analyzing environment
Recurrent models					
Altche[3]	Vehicle trajectory	LSTM	NGSIM	Trajectory prediction on highways, suitable to detect sudden lane changes and prediction upto 10 seconds	Delay in prediction causes missing of some values
Alahi et al.[2]	Pedestrian trajectory	Social-LSTM	ETH,UCY	LSTM model that can predict long-term human trajectory in a scene	Does not consider for different pedestrians with variable locations
Xu et al.[88]	Pedestrian trajectory	Crowd interaction deep neural network (CIDNN)	ETH, UCY, CUHK crowd dataset	Crowded scenarios trajectory prediction with higher accuracy in predictions	More time requirement for working the model
Chan et al.[9]	Vehicle trajectory	Dynamic-Spatial-Attention RNN	Dashcam accident dataset, KITTI	Early and accurate accident anticipation model	Method fails for very complex traffic datasets
Deo et al. [13]	Vehicle trajectory	LSTM encoder-decoder	NGSIM US 101, I-80	Usage of improved social pooling layer instead of sully connected layer	Less accuracy in prediction due to usage of only vehicle tracks to get maneuver class

Table 4.1: Summary of methods, datasets used for trajectory prediction

Reference	Type of prediction	Method used	Dataset	Contribution	Drawback
Su et al.[75]	Pedestrian trajectory	Social-aware LSTM	CUHK crowd, subway station dataset	Crowded scenes accurate near future trajectory prediction	Only considers neighboring positions pf pedestrians, but not in different scenarios
Manh et al.[51]	Pedestrian trajectory	Scene-LSTM	ETH, UCY	LSTM based model to predict human trajectory from video sequences as input	Less prediction quality
Xue et al.[89]	Pedestrian trajectory	Hierarchical LSTM(Social-Scene-LSTM)	ETH, UCY, Town centre dataset	Crowded scene pedestrian trajectory prediction	Prediction accuracy is less for long trajectory predictions
Messaoud et al.[53]	Vehicle trajectory	Relational RNN encoder-decoder	HighD dataset[39]	Long-term prediction upto 5seconds on highway	The metrics used for evaluation are not enough
Scheel et al.[69]	Vehicle trajectory	Bidirectional RNN	NGSIM	Recurrent model based approach that can solve the issues with lane changes in highways	Less accuracy in prediction
Xin et al.[86]	Vehicle trajectory	Intention-aware dual LSTM	NGSIM US 101, I-80	Long-term prediction in highways with higher precision and accuracy	Only suitable for a fixed trajectory
Bartoli et al.[4]	Pedestrian trajectory	Context-aware LSTM	UCY dataset	Prediction model that is able to provide information about person and surroundings	Not suitable for complex scenarios
Kim et al.[36]	Vehicle trajectory	LSTM with 2 layer cell	Real-coordinates obtained from vehicle sensors	LSTM model that is able of predicting complex dynamic future environments	Less prediction accuracy for long-term prediction

Table 4.1: Summary of methods, datasets used for trajectory prediction

Reference	Type of prediction	Method used	Dataset	Contribution	Drawback
Deo et al.[14]	Vehicle trajectory	Maneuver-LSTM	NGSIM US 101	Multi-model trajectory prediction of freeways	Less prediction accuracy for long-term predictions
Combined model (feedforward with recurrent model)					
Xie G et al.[85]	Vehicle trajectory	Sequential model with CNN-LSTM	NGSIM	Accurate and real-time prediction	-
Gaussian Process models					
Goli et al.[28]	Vehicle trajectory	GPR	Lankershim Boulevard	Framework for learning GPR model based on historical data and then use GPR model to predict trajectory	Not clearly explain whether collision is avoided
Wiest et al.[84]	Vehicle trajectory	Gaussian Mixture Model (GMM) and Variational GMM	Differential GPS (DGPS)[29]	Accurate long-term prediction model	-

Table 4.1: Summary of methods, datasets used for trajectory prediction

5 Discussion

There have been different approaches in fields of action anticipation, video prediction and trajectory prediction. In this chapter, a comparison is provided based on method, metrics, datasets used and finally discuss how to apply these methods for anomaly detection tasks.

5.1 Anticipation

5.1.1 Methods used

To choose the best method for anticipation depends on the input data, application, metrics, output and datasets used. Recurrent models are the most chosen network architecture by trajectory prediction and action anticipation, whereas in the case of video prediction, the usage of method is application-specific and can be a recurrent model or generative models. The recurrent models are used for long-term trajectory predictions, but they have less accuracy in predictions. The usage of generative models in trajectory predictions improve accuracy and provides more realistic predictions. Recurrent models are mostly preferred for action anticipation because these models are suitable to handle complex data [18]. The feedforward and generative model-based approaches for video prediction are less accurate and not suitable for real-time scenarios. Recurrent based models are most suitable for real-world video frame prediction applications but lack accuracy. The combination of recurrent and feedforward models can be used for accurate real-time video frame predictions. The most common approach for action anticipation tasks is RNN or any RNN variations. LSTM are the most preferred recurrent model in trajectory predictions. Most of the recurrent model-based approaches are used for early, long-term action anticipations, but they have lesser accuracy for complex scenarios. The video prediction models not only predict the future frames but also detects the anomalies in the prediction. Generative models, for example, GAN are the most common approach in video prediction because of their ability to learn and then generate data similar to the input. In trajectory predictions, delays in the prediction caused by LSTM networks are solved using GAN architecture. In an overall view, recurrent models are mostly preferred for trajectory prediction, action anticipation and recurrent or generative models are preferred for video prediction applications.

Type of anticipation	Method	Application	Drawback
Action anticipation	Feedforward architectures	Long and short-term prediction	Not suitable for dense scenarios
	Recurrent models	Early action anticipation	Less accuracy
	Generative models	Long-term realistic predictions	Not suitable for real-world scenarios
	Combined models	Long-term predictions	-
Video prediction	Feedforward architectures	Long and short-term prediction	Not suitable for dense real-world scenarios
	Recurrent models	Long-term prediction	Less accuracy
	Generative models	Accurate predictions	Not suitable for real-world complex scenarios
Trajectory prediction	Feedforward architectures	Immediate prediction	Not suitable for real-world scenarios
	Recurrent models	Long-term crowded scenarios	Less accuracy
	Gaussian models	Accurate long-term predictions	Not always gives correct predictions
	Combined models	Accurate real-time scenarios	-

Table 5.1: Summary of all methods used for anticipation based on type of prediction

5.1.2 Inputs and Outputs

In every application, the input to the prediction model is historical observation. For video predictions, the input is mostly RGB images and the output is a predicted sequence of frames. In trajectory predictions, the input to the prediction model is historical trajectory data defined in terms of vehicle coordinates. The output of the trajectory prediction model is a predicted vehicle or pedestrian trajectory. The input to an action anticipated model is an image or sequence of actions performed in various environments. The output from the model is an anticipated action label.

5.1.3 Datasets

Dataset used are also application-specific. The most common action anticipation datasets include UCF101, UT-Interaction, JAAD, Breakfast, 50Salads, Human 3.6M, Epic-Kitchen, TVSeries, THUMOS, TV-Human interaction. Action anticipation dataset contains RGB frames of actions performed in various scenarios. Due to unavailability of action anticipation datasets, the datasets used are action recognition datasets. Video prediction dataset includes Human 3.6M, KTH, CUHK, KITTI, BAIR robot pushing dataset, JAAD, Penn Action, UCSD Pedestrian datasets. Some of the datasets are commonly used for action anticipation and video prediction such as UCF101, Human 3.6M, THUMOS. Datasets used for trajectory prediction include NGSIM US-101, NGSIM I-80, Lankershim Boulevard, ETH, UCY, Town centre datasets. The trajectory prediction dataset is again based on the human trajectory or vehicle trajectory application. Some datasets are used in common for action anticipation and trajectory prediction such as Human 3.6M, UCF101.

5.1.4 Metrics

Metrics used for evaluation depends on the application. The model evaluation is performed using metrics namely anticipation accuracy, precision, recall, F1-score for action anticipation. Video prediction metrics are based on evaluating the image quality of predicted frames such as SSIM, PSNR, MSE. The evaluation metrics for trajectory prediction are based on the displacement of the vehicle from a position and predicted trajectory. These metrics include ADE, FDE, RMSE.

5.2 Anticipation for Anomaly Detection

Anomaly detection is the process of detecting deviations of outcomes from nominal or expected outcomes. Various machine learning approaches can be applied for anomaly detection. The basic concept of using anticipation methods for anomaly detection are discussed in this section.

In case of video prediction, initially, the video clips are provided as the input to the anticipation model. Anticipation model aims to anticipate the future frames based on these past observations. The nominal output from this anticipation model is compared with the observed data to determine the predicted value is nominal or anomalous.

Anomaly detection is important in the case of autonomous systems because by detecting anomalies in a future predicted data, the safety of these systems can be ensured. The existing surveys on anomaly detection are mostly application-specific. Most of the existing anomaly detection approaches are used for fraud detection, medical field, intrusion-detection systems, video-surveillance systems. These existing anomaly detection approaches do not solve the issues in autonomous systems. Anomaly detection approaches are suitable to detect and remove the abnormalities from certain observations. The traditional methods for anomaly detection require labelled data, which is difficult to obtain due to the diverse nature. For real-world applications, the anomaly detection methods should have the ability to process high dimensional data. Huge anomalous data is needed for training a model for detecting the anomalous nature of the data.

The application of anticipation methods for anomaly detection possesses some limitations. The anomaly detection methods should process high dimensional data. There is a lack of enough anomalous datasets for training the model. The anomalies are diverse in nature and their detection is furthermore complicated. The concept of anomaly detection also varies based on the application. For example, a fluctuation in readings can be an anomaly in terms of the medical field, but these fluctuations may or may not be an anomaly in case of autonomous systems. The availability of datasets for anomaly detection along with prediction is not yet solved. It is difficult to include all the unexpected behaviours in a dataset.

Most of the deep learning-based approaches are more suitable for anomaly detection tasks, because of their ability to process high dimensional data and to learn the features from the data with or without human intervention [10]. Supervised learning approaches for anomaly detection requires labeled data, which is difficult to obtain with anomalous data.

The various approaches for anomaly detection include neural network, Gaussian models, Bayesian networks, recurrent models. The advantage of using a neural network is that they can be used for binary and multi-class classification. In the case of feedforward architectures, initially, the network is trained only based on nominal behaviours. During the testing phase, the network is provided with a new set of values or instances. The output from this network, whose values are similar to the expected are considered as a normal otherwise abnormal or anomalous event. This comparison is done by calculating the reconstruction error of the output obtained. The reconstruction error is calculated as,

$$\delta_i = \frac{1}{n}\sum_{j=1}^{n}(x_{ij} - o_{ij})^2[10] \tag{5.1}$$

Where δ_i is the reconstruction error and x_i is the test instance, o_i is the output [10]. The higher value of reconstruction error shows more number of anomalous events in the predicted output.

For detecting anomalies in multi-class data, Bayesian networks can be used. The Bayesian networks calculate the anomalies based on posterior probability between the normal and anomalous test data. In this case, the highest posterior probability value is taken as the predicted value.

Another approach for anomaly detection is to use gaussian models, in which the anomalies are calculated based on Maximum Likelihood Estimates (MLE), in which the score is calculated as the difference between the input data and the mean of the data. Once these anomaly scores are obtained, a threshold value is compared to these values, and anomalies are obtained as a result. All of the above-discussed approaches are more suitable for detecting anomalies in various fields. But these approaches are mainly concentrated on detecting anomalies in the predicted data. But these approaches cannot be used along with anticipation and anomaly detection.

The most common approach for long-term anticipation and anomaly detection is to use autoencoders and variations of autoencoders. Autoencoders are the most suitable approach, because of their requirement for unlabeled data for anomaly detection. The features from the data are learned using encoder and decoder, that forms the basic components of autoencoder network. Data sequence is given as the encoder input. The feature vector holding features information representing the input is outputted as the result. Now this feature vector (context vector) is given as the input to the decoder, which reconstructs the input. The output from the model is then compared to the nominal input to detect the anomalies. This comparison is made using reconstruction loss or error as mentioned above. The autoencoders thus can be used for anomaly detection based on how much amount of data being provided as the input is reconstructed. Therefore autoencoders serves as a most viable method for anomaly detection. The variations of the autoencoders such as LSTM, deep autoencoders can also be used for anomaly detection with only change in the way of calculating reconstruction loss. To conclude, autoencoder based approaches are more suitable for anticipation along with anomaly detection. For video anomaly detection, encoder-decoder using LSTM [52], spatio-temporal autoencoder [92] can be used.

6 Conclusion

Anticipation is used in autonomous systems, human-robot interaction, video-surveillance, self-driving cars. Anticipation along with anomaly detection helps in more accurate predictions. This research work identifies the best approach for anomaly detection based on the various types of anticipation approaches provided.

6.1 Contributions

- A comprehensive survey of various anticipation approaches are classified in terms of the type of anticipation.
- The best approach for anticipation in terms of action anticipation, trajectory prediction, video prediction is provided.
- A conclusion about the best model based on the application for long-term, complex scenarios is made.
- Various approaches used for anomaly detection are discussed.
- Finally the best approach for anticipation along with anomaly detection is given.

6.2 Lessons learned

From this survey on video-based anticipation for anomaly detection, most common approaches used for various anticipation methods along with anomaly detection is learned. Feedforward architectures and generative models are suitable for long-term predictions, but not suitable for dense real-world scenarios. For early long-term predictions of crowded scenarios, recurrent models can be applied, but they are less accurate in predictions. Combination of recurrent models and feedforward architecture or recurrent model and generative model is the best option for accurate long-term anticipations. The autoencoders or autoencoder-based approaches are the most suitable for anticipation along with anomaly detection.

6.3 Future work

There still exist difficulties in anticipating the future and detecting anomalies in the predicted output due to the anomalous nature of data points. There is a lack of approaches to exactly predict the output in real-time applications. More evaluation for the application in complex scenarios needs to be performed. There is still a lack of evaluation metrics and datasets needed for anticipated anomaly detection.

References

[1] Yazan Abu Farha, Alexander Richard, and Juergen Gall. When will you do what? - anticipating temporal occurrences of activities. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[3] F. Altché and A. de La Fortelle. An lstm network for highway trajectory prediction. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 353–359, 2017.

[4] F. Bartoli, G. Lisanti, L. Ballan, and A. Del Bimbo. Context-aware trajectory prediction. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1941–1946, 2018.

[5] Alex Berg, Jia Deng, and L Fei-Fei. Imagenet large scale visual recognition challenge 2010, 2010.

[6] Hakan Bilen, Basura Fernando, Efstratios Gavves, and Andrea Vedaldi. Action recognition with dynamic image networks, 2017.

[7] Lluis Castrejon, Nicolas Ballas, and Aaron Courville. Improved conditional vrnns for video prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7608–7617, 2019.

[8] Antoni Chan and Nuno Vasconcelos. Ucsd pedestrian dataset. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(5):909–926, 2008.

[9] Fu-Hsiang Chan, Yu-Ting Chen, Yu Xiang, and Min Sun. Anticipating accidents in dashcam videos. In *Asian Conference on Computer Vision*, pages 136–153. Springer, 2016.

[10] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), July 2009. ISSN 0360-0300.

[11] Benjamin Coifman and Lizhe Li. A critical evaluation of the next generation simulation (ngsim) vehicle trajectory dataset. *Transportation Research Part B: Methodological*, 105(C):362–377, 2017.

[12] Dima Damen, Hazel Doughty, Giovanni Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Computer Architecture Letters*, (01):1–1, 2020.

[13] Nachiket Deo and Mohan M. Trivedi. Convolutional social pooling for vehicle trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

[14] Nachiket Deo and Mohan M Trivedi. Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1179–1184. IEEE, 2018.

[15] Nuno Ferreira Duarte, Mirko Raković, Jovica Tasevski, Moreno Ignazio Coco, Aude Billard, and José Santos-Victor. Action anticipation: Reading the intentions of humans and robots. *IEEE Robotics and Automation Letters*, 3(4):4132–4139, 2018.

[16] Hehe Fan, Linchao Zhu, and Yi Yang. Cubic lstms for video prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8263–8270, 2019.

[17] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, pages 64–72, 2016.

[18] A. Furnari and G. M. Farinella. Egocentric action anticipation by disentangling encoding and inference. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3357–3361, 2019.

[19] Enric Galceran, Alexander G Cunningham, Ryan M Eustice, and Edwin Olson. Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction. In *Robotics: Science and Systems*, volume 1, 2015.

[20] Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Predicting the future: A jointly learnt model for action anticipation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[21] Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Forecasting future action sequences with neural memory networks. *arXiv preprint arXiv:1909.09278*, 2019.

[22] Hang Gao, Huazhe Xu, Qi-Zhi Cai, Ruth Wang, Fisher Yu, and Trevor Darrell. Disentangling propagation and generation for video prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[23] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. RED: reinforced encoder-decoder networks for action anticipation. *CoRR*, abs/1707.04818, 2017.

[24] Roeland De Geest, Efstratios Gavves, Amir Ghodrati, Zhenyang Li, Cees Snoek, and Tinne Tuytelaars. Online action detection, 2016.

[25] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[26] Shilpa Gite and Himanshu Agrawal. Early prediction of driver's action using deep neural networks. *International Journal of Information Retrieval Research (IJIRR)*, 9(2):11–27, 2019.

[27] Shilpa Gite, Himanshu Agrawal, and Ketan Kotecha. Early anticipation of drivers maneuver in semiautonomous vehicles using deep learning. *Progress in Artificial Intelligence*, 8, 03 2019.

[28] Sepideh Afkhami Goli, Behrouz H Far, and Abraham O Fapojuwo. Vehicle trajectory prediction with gaussian process regression in connected vehicle environment. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 550–555. IEEE, 2018.

[29] Christoph Hermes, Jürgen Wiest, Christian Wöhler, Ulrich Kreßel, and Franz Kummert. Manifold-based motion prediction. *Proc. 6. Dortmunder Auto-Tag. Dortmund, Germany*, 2011.

[30] Alejandro Hernandez, Jurgen Gall, and Francesc Moreno-Noguer. Human motion prediction via spatio-temporal inpainting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7134–7143, 2019.

[31] A. Horé and D. Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th International Conference on Pattern Recognition*, pages 2366–2369, 2010.

[32] Matin Hosseini, Anthony S Maida, Majid Hosseini, and Gottumukkala Raju. Inception-inspired lstm for next-frame video prediction. *arXiv preprint arXiv:1909.05622*, 2019.

[33] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014.

[34] Ashesh Jain, Avi Singh, Hema S Koppula, Shane Soh, and Ashutosh Saxena. Recurrent neural networks for driver activity anticipation via sensory-fusion architecture. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3118–3125. IEEE, 2016.

[35] Qiuhong Ke, Mario Fritz, and Bernt Schiele. Time-conditioned action anticipation in one shot. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9925–9934, 2019.

[36] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi. Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 399–404, 2017.

[37] Yu Kong and Yun Fu. Human action recognition and prediction: A survey, 2018.

[38] Yu Kong, Zhiqiang Tao, and Yun Fu. Deep sequential context networks for action prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[39] Robert Krajewski, Julian Bock, Laurent Kloeker, and Lutz Eckstein. The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2118–2125. IEEE, 2018.

[40] Yong-Hoon Kwon and Min-Gyu Park. Predicting future frames using retrospective cycle gan. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1811–1820, 2019.

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

[42] 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 European Conference on Computer Vision (ECCV)*, September 2018.

[43] Yuke Li. Which way are you going? imitative decision learning for path forecasting in dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[44] Xiaodan Liang, Lisa Lee, Wei Dai, and Eric P. Xing. Dual motion gan for future-flow embedded video prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[45] Chunhui Liu, Yueyu Hu, Yanghao Li, Sijie Song, and Jiaying Liu. Pku-mmd: A large scale benchmark for continuous multi-modal human action understanding. *arXiv preprint arXiv:1703.07475*, 2017.

[46] Jun Liu, Amir Shahroudy, Gang Wang, Ling-Yu Duan, and Alex C. Kot. Ssnet: Scale selection network for online 3d action prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[47] Wen Liu, Weixin Luo, Dongze Lian, and Shenghua Gao. Future frame prediction for anomaly detection – a new baseline. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6536–6545, 2018.

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

[49] Cewu Lu, Jianping Shi, and Jiaya Jia. Abnormal event detection at 150 fps in matlab. In *Proceedings of the IEEE international conference on computer vision*, pages 2720–2727, 2013.

[50] Srikanth Malla, Behzad Dariush, and Chiho Choi. Titan: Future forecast using action priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11186–11196, 2020.

[51] Huynh Manh and Gita Alaghband. Scene-lstm: A model for human trajectory prediction. *arXiv preprint arXiv:1808.04018*, 2018.

[52] Jefferson Ryan Medel and Andreas Savakis. Anomaly detection in video using predictive convolutional long short-term memory networks, 2016.

[53] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet, and F. Nashashibi. Relational recurrent neural networks for vehicle trajectory prediction. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1813–1818, 2019.

[54] Antoine Miech, Ivan Laptev, Josef Sivic, Heng Wang, Lorenzo Torresani, and Du Tran. Leveraging the present to anticipate the future in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[55] Nishant Nikhil and Brendan Tran Morris. Convolutional neural network for trajectory prediction. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, September 2018.

[56] Marc Oliu, Javier Selva, and Sergio Escalera. Folded recurrent neural networks for future video prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 716–731, 2018.

[57] Sergiu Oprea, Pablo Martinez-Gonzalez, Alberto Garcia-Garcia, John Alejandro Castro-Vargas, Sergio Orts-Escolano, Jose Garcia-Rodriguez, and Antonis Argyros. A review on deep learning techniques for video prediction. *arXiv preprint arXiv:2004.05214*, 2020.

[58] A. Patron-Perez, M. Marszalek, I. Reid, and A. Zisserman. Structured learning of human interactions in tv shows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2441–2453, 2012.

[59] Fiora Pirri, Lorenzo Mauro, Edoardo Alati, Valsamis Ntouskos, Mahdieh Izadpanahkakhk, and Elham Omrani. Anticipation and next action forecasting in video: an end-to-end model with memory. *arXiv preprint arXiv:1901.03728*, 2019.

[60] Amir Rasouli. Deep learning for vision-based prediction: A survey. *arXiv preprint arXiv:2007.00095*, 2020.

[61] Amir Rasouli, Iuliia Kotseruba, and John K. Tsotsos. Are they going to cross? a benchmark dataset and baseline for pedestrian crosswalk behavior. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[62] Amir Rasouli, Iuliia Kotseruba, and John K Tsotsos. Pedestrian action anticipation using contextual feature fusion in stacked rnns. *arXiv preprint arXiv:2005.06582*, 2020.

[63] Fitsum A. Reda, Guilin Liu, Kevin J. Shih, Robert Kirby, Jon Barker, David Tarjan, Andrew Tao, and Bryan Catanzaro. Sdcnet: Video prediction using spatially-displaced convolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[64] Cristian Rodriguez, Basura Fernando, and Hongdong Li. Action anticipation by predicting future dynamic images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[65] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M Kitani, Dariu M Gavrila, and Kai O Arras. Human motion trajectory prediction: A survey. *The International Journal of Robotics Research*, 39 (8):895–935, 2020.

[66] Michael S Ryoo and JK Aggarwal. Ut-interaction dataset, icpr contest on semantic description of human activities (sdha). In *IEEE International Conference on Pattern Recognition Workshops*, volume 2, page 4, 2010.

[67] Mohammad Sadegh Aliakbarian, Fatemeh Sadat Saleh, Mathieu Salzmann, Basura Fernando, Lars Petersson, and Lars Andersson. Encouraging lstms to anticipate actions very early. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[68] K. Saleh, M. Hossny, and S. Nahavandi. Real-time intent prediction of pedestrians for autonomous ground vehicles via spatio-temporal densenet. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9704–9710, 2019.

[69] O. Scheel, L. Schwarz, N. Navab, and F. Tombari. Situation assessment for planning lane changes: Combining recurrent models and prediction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2082–2088, 2018.

[70] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 32–36 Vol.3, 2004.

[71] Paul Schydlo, Mirko Rakovic, Lorenzo Jamone, and José Santos-Victor. Anticipation in human-robot cooperation: A recurrent neural network approach for multiple action sequences prediction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–6. IEEE, 2018.

[72] Yuge Shi, Basura Fernando, and Richard Hartley. Action anticipation with rbf kernelized feature mapping rnn. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 301–317, 2019.

[73] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *Advances in neural information processing systems*, 27:568–576, 2014.

[74] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild, 2012.

[75] Hang Su, Jun Zhu, Yinpeng Dong, and Bo Zhang. Forecast the plausible paths in crowd scenes. In *IJCAI*, volume 1, page 2, 2017.

[76] Ruben Villegas, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee. Learning to generate long-term future via hierarchical prediction. *arXiv preprint arXiv:1704.05831*, 2017.

[77] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating visual representations from unlabeled video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 98–106, 2016.

[78] Vedran Vukotić, Silvia-Laura Pintea, Christian Raymond, Guillaume Gravier, and Jan C Van Gemert. One-step time-dependent future video frame prediction with a convolutional encoder-decoder neural network. In *International Conference on Image Analysis and Processing*, pages 140–151. Springer, 2017.

[79] Chujie Wang, Lin Ma, Rongpeng Li, Tariq S Durrani, and Honggang Zhang. Exploring trajectory prediction through machine learning methods. *IEEE Access*, 7:101441–101452, 2019.

[80] D. Wang, Y. Yuan, and Q. Wang. Early action prediction with generative adversarial networks. *IEEE Access*, 7:35795–35804, 2019.

[81] Hongsong Wang and Jiashi Feng. Delving into 3d action anticipation from streaming videos. *arXiv preprint arXiv:1906.06521*, 2019.

[82] Yunbo Wang, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li Fei-Fei. Eidetic 3d lstm: A model for video prediction and beyond. In *International Conference on Learning Representations*, 2018.

[83] Nevan Wichers, Ruben Villegas, Dumitru Erhan, and Honglak Lee. Hierarchical long-term video prediction without supervision. *arXiv preprint arXiv:1806.04768*, 2018.

[84] Jürgen Wiest, Matthias Höffken, Ulrich Kreßel, and Klaus Dietmayer. Probabilistic trajectory prediction with gaussian mixture models. In *2012 IEEE Intelligent Vehicles Symposium*, pages 141–146. IEEE, 2012.

[85] Guo Xie, Anqi Shangguan, Rong Fei, Wenjiang Ji, Weigang Ma, and Xinhong Hei. Motion trajectory prediction based on a cnn-lstm sequential model. *Science China Information Sciences*, 63(11):1–21, 2020.

[86] Long Xin, Pin Wang, Ching-Yao Chan, Jianyu Chen, Shengbo Eben Li, and Bo Cheng. Intention-aware long horizon trajectory prediction of surrounding vehicles using dual lstm networks. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1441–1446. IEEE, 2018.

[87] Mingze Xu, Mingfei Gao, Yi-Ting Chen, Larry S Davis, and David J Crandall. Temporal recurrent networks for online action detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5532–5541, 2019.

[88] Yanyu Xu, Zhixin Piao, and Shenghua Gao. Encoding crowd interaction with deep neural network for pedestrian trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5275–5284, 2018.

[89] Hao Xue, Du Q Huynh, and Mark Reynolds. Ss-lstm: A hierarchical lstm model for pedestrian trajectory prediction. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1186–1194. IEEE, 2018.

[90] Yufei Ye, Maneesh Singh, Abhinav Gupta, and Shubham Tulsiani. Compositional video prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10353–10362, 2019.

[91] Shuai Yi, Hongsheng Li, and Xiaogang Wang. Pedestrian behavior understanding and prediction with deep neural networks. In *European Conference on Computer Vision*, pages 263–279. Springer, 2016.

[92] Yiru Zhao, Bing Deng, Chen Shen, Yao Liu, Hongtao Lu, and Xian-Sheng Hua. Spatio-temporal autoencoder for video anomaly detection. In *Proceedings of the 25th ACM International Conference on Multimedia*, page 1933–1941. Association for Computing Machinery, 2017. ISBN 9781450349062.