Predicting pedestrian path using optical flow as context cue

Abhishek Tomy

Predicting pedestrian path using optical flow as context cue

by

Abhishek Tomy

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday October 26, 2020 at 11:00 AM.

Student number:	4771729	
Project duration:	January, 2020– October,2020	
Thesis committee:	Prof. dr. D. Gavrila,	TU Delft, supervisor
	Ir. E. A. I. Pool	TU Delft,supervisor
	Dr. L. Ferranti	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Abstract

A human driver can gauge the intention and signals given by other road users indicative of their future behaviour. The intentions and signals are identified by looking at the cues originating from vulnerable road users or their surroundings (hand signals, head orientation, posture, traffic signals, distance to curb, etc.). Taking all these cues into account by creating a separate detector for each is an extremely difficult task. Instead, this MSc Thesis will explore the possibility of using a generic contextual cue in optical flow originating from a pedestrian with deep learning methods to improve the path prediction in a naturalistic driving scenario.

The contribution of this work is to examine multiple ways to extract relevant information from the optical flow and also explore the possibility of using the entire the high-dimensional optical flow using convolutions and soft-attention to help identify relevant pixels for the prediction task. This work elaborates on the extraction and processing of optical flow features. It proposes 2 Recurrent neural networks (RNN) based model: one to work with the histogram of optical flow features and the other one to take in the dense optical flow directly. Also, visualisation of the soft-attention weights is done to add a step that helps in the interpretability of the RNN model incorporating dense optical flow. From the experimental results, optical flow features have shown significant improvements in terms of predicting probabilistic confidence for tracks with some changes in their motion mode. It was seen that the convolution-attention RNN model was able to work with dense optical flow features and position of pedestrians as input to obtain better results among all the combinations of features and models compared in this work.

Contents

Acknowledgements

This thesis represents the result of the MSc project on path prediction of a pedestrian in a naturalistic driving scenario which I undertook with the Intelligent Vehicles group at TU Delft. First of all, I would like to thank my supervisor's Ewoud Pool and Prof. Dariu Gavrila for their guidance and support. I am extremely grateful to Dariu who had given me the opportunity to work on this very interesting project and the opportunity to work closely with the Intelligent Vehicles group that is at the forefront of ensuring safe interaction of the autonomous vehicle with the vulnerable road users. His guidance, recommendations, and the structure that he bought to the organisation of the thesis work has been really helpful and ensured a smooth process. I have to thank him for the clear and strict emphasis on certain points and expectations during the meetings which have helped act as pointers for me later on during the experimentation. His accommodation of my requests and even extremely prompt responses have ensured that I had to worry only about my thesis work and nothing else. I would like to express my gratitude and appreciation to Ewoud Pool who has been actually extremely patient with me, discussion every week and even on unplanned days has helped me in having clarity and understanding of the subject, and also his pointed questions helped to keep me on track with the thesis work whenever I was going wrong. His right guidance helped in rectifying the mistakes with constructive and positive criticisms. I am glad for all his support, freedom, and appreciation throughout the project. I would also like to extend my appreciation to Sebastian Krebs at Daimler who helped me by providing with all the information and data that I requested for working with the dataset that I used to conduct my experiments. He had taken out time from his busy schedule to respond to my requirements and I would like to thank him again for his time and effort that ensured that I had everything that I needed.

I would like to thank Divya and Eldho for all the weekend plans, movies, and dinners because of which Delft felt the same as home. I would like to thank the friends with whom these two years went like a breeze and for all the fun conversations and being there always.

Additionally, I am extremely grateful to my parents, who have shown great confidence in me and their unconditional and well-needed support. For they were always a call away and ensured that these two years, I have never felt that I am far-away from home.

Abhishek Tomy
Delft, October 2020

1

Introduction

Autonomous driving is one of the key technologies that will shape the future of mobility. It will facilitate easy access to mobility and the possibility of cheap and convenient transportation, with fewer people owning a personal car. Few of the possibilities for the future include a subscription type model to mobility with fleet operators providing transportation as a service. The subscription type model by calling the vehicle according to ones need would enable efficient use of the vehicle and public spaces that otherwise would be occupied for parking. Also, it can improve public transportation by providing last-mile connectivity. Autonomous Vehicle (AV) will allow for a more leisurely time while travelling. It can also increase productivity by allowing the passenger to occupy themselves with professional or learning tasks while being driven to work. One of the major advantages would be that autonomous driving will reduce accidents caused by human error and fatigue.

The work in this thesis delves deeper into one of the challenges of autonomous driving when it has to operate in a space co-occupied by Vulnerable Road User (VRU). This chapter is organised as follows: section 1.1 introduces the various automation levels as defined by Society of Automotive Engineers (SAE). This will help in distinguishing and having a clear definition of the autonomous driving challenges and possibilities. section 1.2 elaborates on environment perception, which is vital to make autonomous driving a reality. section 1.3 explains about the path prediction task and why it is essential for the safe interaction of the autonomous vehicle and a VRU. Finally, in section 1.4, the research questions that this work tries to address are elaborated.

1.1. Automation levels

To understand the current capabilities and to measure the progress in technology of AV, SAE has defined 6 levels of driving automation in its J3016 standard [29] ranging from SAE Level-0 (no automation) to SAE Level-5 (fully autonomous) as shown in fig. 1.1. In Level-0, the sole responsibility of driving task falls on the driver with no automation, with the subsystem only existing to provide information and warnings. Under Level-1, one of the driving tasks such as lane-keeping or maintaining longitudinal speed etc. is automated, and the driver is always in the loop for every driving task and the sole responsibility of handling a developing situation rests with the driver. A Level-2 system will have more than one control function automated. Here, a vehicle can control both the lateral (steering) and longitudinal (speed/braking) motion of the vehicle simultaneously along with the driver paying constant attention and taking care of all remaining aspects of

the dynamic driving task. In a Level-3 system, the vehicle can perform the dynamic driving task under certain conditions without supervision. The vehicle will be able to detect and react to its surrounding, therefore allowing hands-free driving. However, the driver is expected to be awake and in a position to take control when the automated system gives a warning. Level- 4 is a high driving automation system that can perform the complete dynamic driving task in pre-defined routes or 'operational domain' in which the vehicle is supposed to run. The driver is not expected to respond to the requests to take back control or respond to any situation if the vehicle is within that specific route. Finally, Level-5 is the full driving automation system that can perform the dynamic driving task in all situations and is independent of any particular operation domain or routes. The driver is completely removed from the loop.

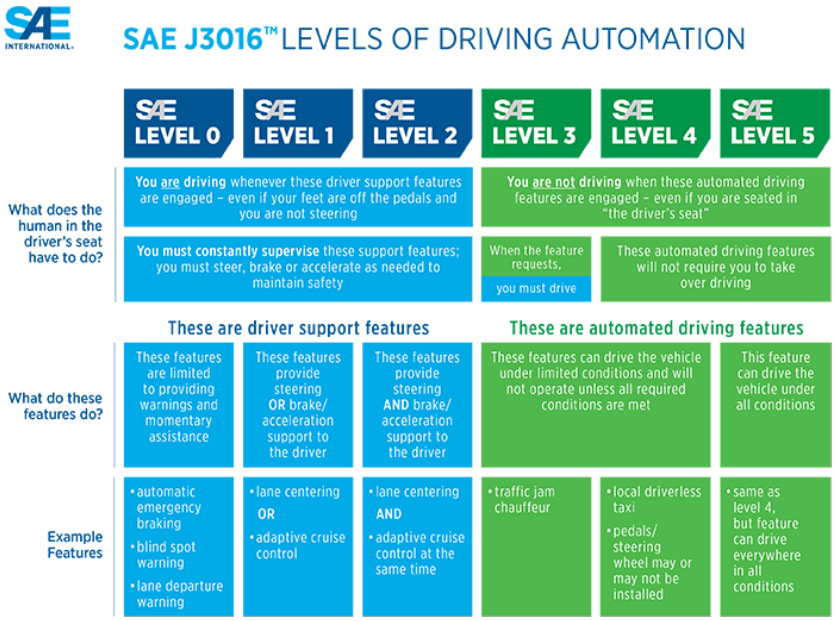

Figure 1.1: Automation levels as defined by SAE [29]

1.2. Environment perception

An autonomous vehicle has to detect and perceive the environment better than humans for acceptance and safe operation. For high levels of autonomy starting from Level-4 where the driver is not expected to take back control, the vehicle should be able to operate in all types of weather and lighting condition without intervention. Apart from this, for a reliable and safe driving, there should be redundancy and fallback for various sensors. An autonomous vehicle needs to be equipped with diverse sensors specializing and complimenting each other in different operating conditions and also as verification for each others perception of the environment. Currently, major sensors under development for environment perception are camera, Light Detection and Ranging (LiDAR), Radio Detection and Ranging (radar) and acoustic sensors. Working with each other, they can detect, classify and estimate position and speed of objects. Apart from these environment perception sensors, the vehicle needs to be equipped with Global Positioning System (GPS) and Inertial measurement unit (IMU) to help provide the ego-vehicle's location and motion in the environment.

Among the environment perception sensors, the camera gives the best visual representation of the environment as it is not just able to detect objects but also recognize them. However, cameras have limitations in low visibility conditions and are also affected by the weather conditions when it is raining or foggy outside. Also, a single camera setup cannot localize the detected objects in the 2-D world without incorporating prior

knowledge. LiDAR, which works on the principle of emitting and reception of the reflected light pulses will enable the autonomous car to have a 3-D view of the entire environment. It can detect the depth and shape of the object, and can work in low-visibility conditions, thereby complimenting the camera sensor. Similar to LiDAR, the radar works on transmission and reflection of radio waves from the object. Depth and speed can be estimated based on the difference between the transmitted and received wave frequencies. One of the advantages of radar is that it can work well in all weather conditions compared to LiDAR or camera which gets affected by rain, fog or snow. Also, compared to LiDAR, radar is relatively less expensive. Among the available sensors, acoustic sensors can play an essential role in detecting objects that are not in the conventional line-of-sight of the sensors mentioned above. It can detect objects approaching from behind an obstacle and can provide early warnings since sound can propagate beyond-line-of-sight of the source. For example, using auditory cues would be helpful to detect the sirens of emergency vehicles in the vicinity and react to them as early as possible.

1.3. Prediction Task

Research in the field of autonomous driving can be divided into 4 component blocks that depend upon each other, as shown in Figure 1.2. The first stage involves detection and identification of objects in the environment using sensors such as camera, radar, LiDAR and acoustic sensors. It is followed by temporal association of observations to existing tracks or creation of new tracks for a previously unseen agent. Tracks and detections of the agents are then fed to the third stage to predict future trajectories or the intention of the agent. Path prediction stage will make use of observations from multiple sensors. Images from the camera will be used to extract visual cues, LiDAR, radar or stereo vision would localize the agent in the ego-vehicle environment and information from GPS, and IMU can be used to provide the localization in a world coordinate. In the final stage, knowledge of the environment from the detection stage and the predictions will be used to determine the available free spaces in the future and path of the AV will be planned accordingly.

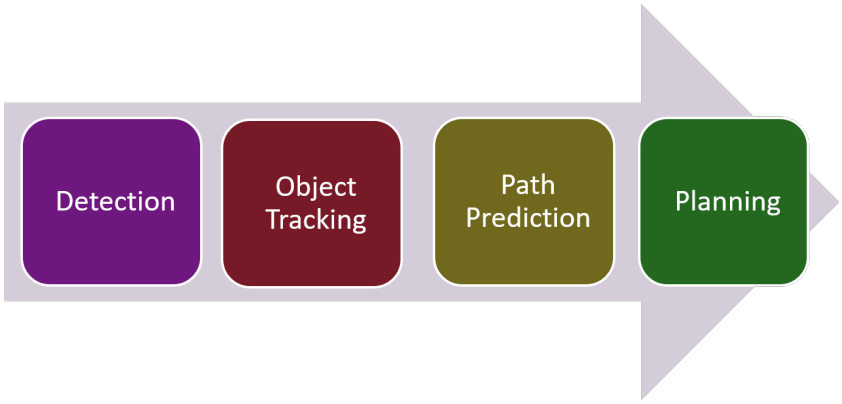

Figure 1.2: Components blocks for autonomous driving

One of the main challenges of Autonomous Driving System (ADS) is to predict and reason how a VRU would behave in the future. Path prediction becomes important as automation levels increase, and the vehicle is expected to operate in complex urban traffic scenarios with constant interaction with multiple VRU. From 'Level-4' on-wards, the AV is expected to handle the complete dynamic driving task without human interventions. With path prediction, an AV will have a glimpse of how the environment would evolve in future time steps allowing that extra fraction of a second to react, ensuring safety and comfort. In a sample scenario shown in fig. 1.3, the AV has to identify whether the pedestrians would stop at the curb or will they cross in front of the vehicle. Path prediction methods that can identify and understand these type of intentions and

prevent dangerous situations. For understanding the intentions of a VRU, additional contextual information such as environment layout, traffic signals, distance to the curb, gestures, awareness or body posture needs to be included to understand what a VRU intends to do. Taking all these cues into account by creating a separate detector for each is an impossible task. Instead, looking at a generic cue that can capture as much as possible information would be ideal. A human driver relies primarily on the visual input for driving. However, based on driving experience, humans are able to pay attention to relevant areas in the scene or other road users that would affect driving. So, instead of using the entire scene as contextual information in the form of a high-dimensional image, the problem can be broken down to extract specific generic cues originating from the environment or other individual road users. As this work will look into short-term trajectory prediction of a pedestrian for safety-critical situations; sudden changes in motion and preceding signals originating from a pedestrian body becomes significant. This work will look into the changes in the image from frame to frame to detect changes in motion behaviour. The difference in location of the image features between two frames is called optical flow and is further explained in section 3.1. The optical flow will implicitly contain similar spatial constraints and shape as images, along with an additional correlation to motion.

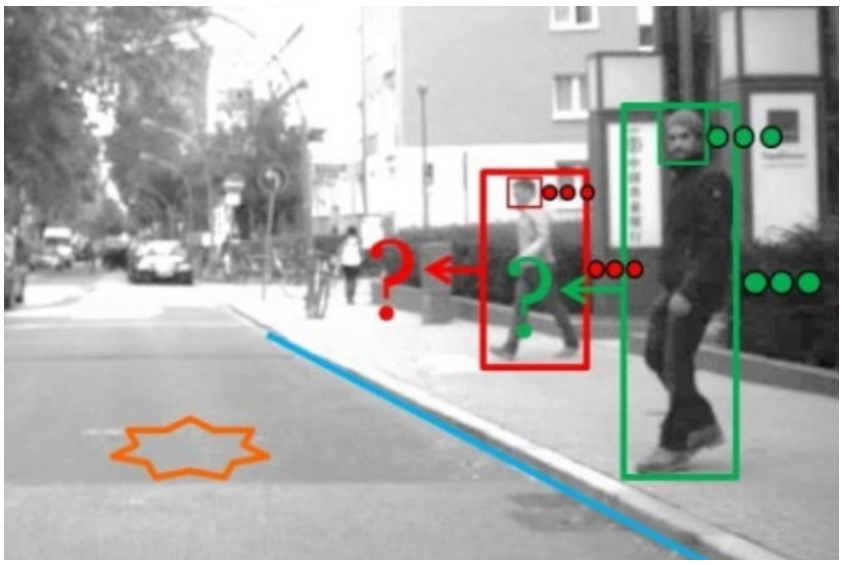

Figure 1.3: Representative scenario for a case when an AV has to identify the crossing intention and predict the path for safe interaction(from [35])

1.4. Research question

The importance of path prediction and how a contextual cue will enable the prediction method to predict VRU behaviour has been discussed in section 1.3. This work will investigate the incorporation of a generic contextual cue in optical flow for the path prediction task. Hence, the main research question of the thesis is: Does the incorporation of optical flow as a contextual cue improve the path prediction performance?

The sub-questions for this investigation are:

- What is a suitable method and approach to model path prediction?
- How does the accuracy in terms of probabilistic confidence and predicted position affected by the incorporation of optical flow?
- Is dimensionality reduction needed to make dense optical flow a feasible contextual cue?

2

Related Work

This chapter is organised to introduce the development and related works in the area of path prediction (section 2.1) and progressively expand and discuss in-depth the methods that have adopted non-linear regression approach using Recurrent Neural Network (RNN) (section 2.2). In section 2.3, various types of contextual cues used for the prediction task are mentioned. In section 2.4, the available datasets and benchmarks for path prediction task are listed down. In section 2.5, the performance of methods on benchmarks is aggregated to evaluate the current state-of-art. Finally, in section 2.6, the contribution of this work along with the choice of model and contextual cues are elaborated by discussing the rationale behind the choices.

2.1. Prediction method taxonomy

Path prediction methods can be broadly divided into 3 categories, as shown in fig. 2.1 [52]: physics-based methods use explicit motion models to make predictions; the planning based approaches minimize the objective function of the path taken to reach a particular goal; and the pattern-based approaches learn the underlying motion pattern from a database. In this section, the previous work in the field of path prediction will be discussed and will be organized into one of these three categories.

2.1.1. Physics based approaches

Physics-based approaches have explicitly defined motion models and can also use multiple motion models that can either be combined or selected based on the estimated state. The future position of the vulnerable road user is then calculated by recursively applying the pre-defined or estimated motion model. Kalman filters (KF) is one of the well known and widely adopted methods that fall under this category. Kalman filters can be used for predictions by recursively propagating the current state with a pre-defined dynamical model such as Constant Turn (CT), Constant Velocity (CV) or Constant Acceleration (CA) [23]. Apart from using explicitly defined motion models, the physics-based approaches also include methods that can make use of a pre-defined set of rules. Helbing and Molnar [24] used the social force model to assign various attractive and repulsive forces to model human-to-human interactions in the crowd. In [2], the social force model was expanded by mapping the affinity of people to move together based on their proximity and their relative position in the neighbourhood of agent.

A VRU may exhibit various motion's (e.g. linear movements, turning, accelerating). In order to capture

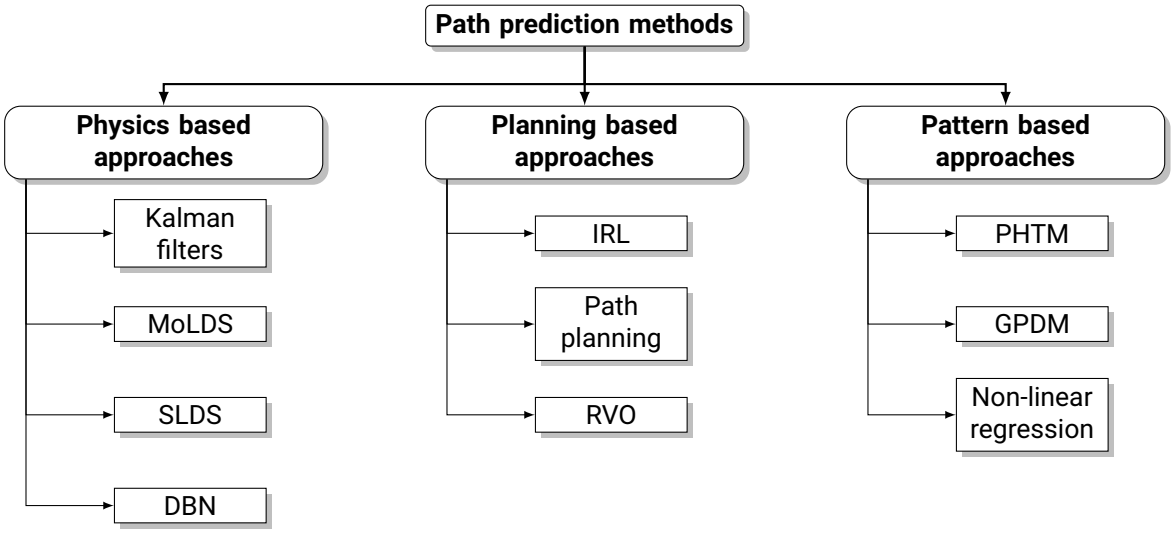

Figure 2.1: Classification of path prediction methods. Under each modelling approaches, few of the methods that fall under those categories are given, which would be elaborated going forward.

these variations, multiple motion models approaches such as Mixture of Linear Dynamical Systems (MoLDS), Switching Linear Dynamical System (SLDS) and Dynamic Bayesian Networks (DBN) are adopted. MoLDS consists of multiple motion models and a suitable model is activated based on priors and observations. In [43], informed and uninformed MoLDS was used to predict the trajectory of a cyclist. The uninformed-MoLDS is initialized with uniform priors whereas the informed-MoLDS includes priors, furthermore, in [43] the local topology was used to assign a prior for the motion models at various locations. Similarly, an SLDS incorporates multiple motion models and uses state parameter to define the probability of being in a particular motion state and a transition probability matrix to define the probability of switching between various states. As observed in [35], an SLDS can acknowledge the change in motion models after sufficient observations contradict the predictions from the currently active dynamic model but cannot anticipate the change even before it occurs. However, DBN have the ability to take in visual or other cues and provide reasoning for the possible cause of change and forecast the VRU behaviour. In [35], the DBN was used to extend the prediction task to a general non-linear probabilistic graphical framework incorporating multiple models and cues such as the distance between pedestrian and ego-vehicle, head orientation and distance of the pedestrian from the curb.

2.1.2. Planning based approaches

Planning-based approaches use an objective function to minimize the cost of a sequence of action, implying, the methods account for the impact of current and future actions on the future state of the model [57]. Planning-based approaches can be sub-divided into two categories based on the cost function: forward-planning and inverse-planning approaches.

Forward-planning approaches use a user-defined cost function to infer motion. One of the approaches is to use a global path planning method to predict a trajectory based on the final goal combined with a local collision avoidance methods such as the Reciprocal Velocity Obstacles (RVO) [61]. In [51], a forward-planning approach, the trajectory is obtained by coupling the planning and prediction part. The prediction part accounts for the conflicting objectives and goals of other agents in the scene, and the resulting cost is used to plan the future trajectory of the agent. Inverse-planning approaches adopt an online estimation to calculate the cost function and one of the ways to estimate the cost function is through the addition of Inverse Rein-

forcement Learning (IRL) or also or also called as Inverse Optimal Control (IOC) module [69]. In [37], an IRL module was added to the RNN network for refining and ranking the trajectories by accounting for agent interactions, semantics and the expected reward. In [34] and [69], semantic map of the environment was used to learn the cost function identifying the preferences of an agent to traverse the environment.

2.1.3. Pattern based approaches

Pattern-based approaches do not require an explicitly defined dynamical motion model. The focus is on designing an effective learning strategy to discover patterns or dependencies from the training data. Pattern-based methods can be distinguished into two categories: clustering-based (e.g. PHTM) and regression-based approaches (e.g. GPDM, RNN).

Probabilistic Hierarchical Trajectory Matching (PHTM) method matches the feature vectors hierarchically to a database of motion trajectories and the corresponding feature vectors. The closest match of the current trajectory to the one in the database is used to extrapolate future motions from the current state. The input vector for the trajectory database can also contain relevant contextual cues and kinematics. In [32], pedestrian state consisted of lateral and longitudinal positions along with low-dimensional optical flow features. However, clustering-based approaches are limited in their successes because of their limitation in large data size and dimensions [6].

Gaussian Process Dynamical Models (GPDM) [64] is a non-linear method that maps dynamics to a low-dimensional latent representation and learns the model dynamics along with the mapping of latent representation to the observation space using Gaussian Process (GP) regression. Once the non-linear mapping between the latent and observation space is learned, future predictions are made in the latent space and then translated to observation space. In [32], GPDM is used for dimensionality reduction of the dense optical flow features to a lower-dimensional learned latent representation. Walking and stopping models are separately trained, and the model is able to predict the trajectory and optical flow for future instances. GPDM can generalize well even with small amount of training samples. However, performance or learning is a significant issue in large datasets.

Non-linear regression approaches learn the implicit dynamics and patterns sequentially from the training data, with the development and progress in machine learning, RNN have shown promising results in recent works involving prediction tasks. This area and the recent works would be further explored in the section 2.2, as these approaches have produced state-of-art results in benchmarks and in recent times, have become increasingly popular for the path prediction task.

2.2. Recurrent Neural Network

An RNN is a type of deep learning method that can handle variable-length input sequences and learn the underlying pattern of time-series or sequences (fig. 2.2). Recent successes of RNN can be attributed to the sophisticated recurrent units such as Gated Recurrent Unit (GRU) [13] and Long Short-Term Memory (LSTM) [25]. These units allow retaining the vital information over a long range of the sequence.

In this section, an in-depth look into some of the recent works that have adapted the RNN to model trajectory prediction is presented. The work has been categorised to understand the underlying design choices and why they were chosen for building the network architecture.

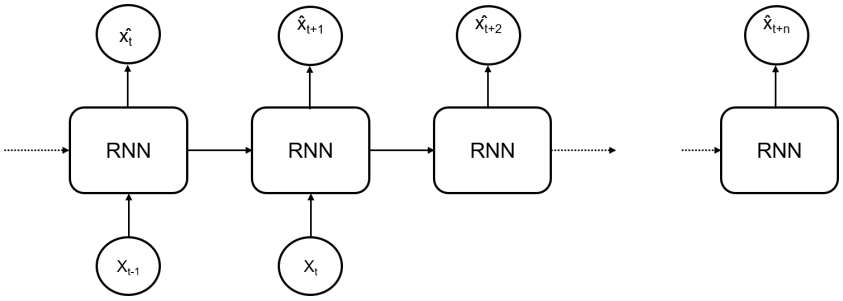

Figure 2.2: A basic RNN architecture that makes future predictions based on past observation. An input in the form x_t is fed to the network at time t, based on previous observations x_{t-i} and the current input, the network can make predictions for $t+n$ steps ahead.

2.2.1. Recurrent unit based approach

RNN based methods have shown that it is possible to model complex social interactions to predict trajectories. In [3], the proposed Social-LSTM use the simple LSTM unit to capture the inter-dependencies of multiple agents in the scene creatively. LSTM have a remarkable ability to learn long sequences; however, they cannot capture co-relation between agents. In the proposed network, individual pedestrian paths are modelled using an LSTM; however, LSTM of individual agents are connected through a social-pooling layer as shown in fig. 2.3. Information about the hidden states of each neighbouring LSTM is shared to reason about future predictions implicitly. The spatial information about the neighbours is preserved in the social-pooling layer. In [22], Social-GAN combines the LSTM with a Generative Adversarial Network. LSTM is used to observe motion-history and a Generative Adversarial Network (GAN) is used to generate socially plausible trajectories by aggregation of information from various layers.

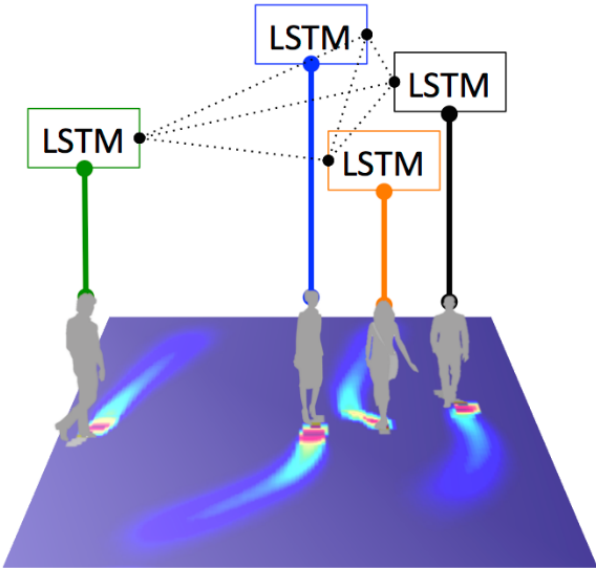

Figure 2.3: Social LSTM: Individual LSTM are connected to each other through a social-pooling layer allowing the sharing of information with spatially proximal LSTM's (from [3])

2.2.2. Convolutional RNN

Images contain a lot of spatial information and constraints and directly supplying it to a vanilla RNN network will lead to loss of vital information. The convolutional RNN architecture was developed to support tasks such as activity recognition or trajectory prediction that takes images as input, machine translation etc. Con-

volutional RNN uses CNN layers as a feature extractor for images and videos, which is then fed to the RNN layers for temporal predictions.

In [56], the proposed INFER network uses an intermediate representation in the form of 5 semantic channels: obstacles, road, lane markings, target vehicle and other vehicles. These semantic channels are independent of the sensing modalities and hence generalises well to different datasets and can work with various type of sensors. Inputs from these multiple channels(fig. 2.4) are then fed to a RNN network to learn the spatial and temporal dynamics.

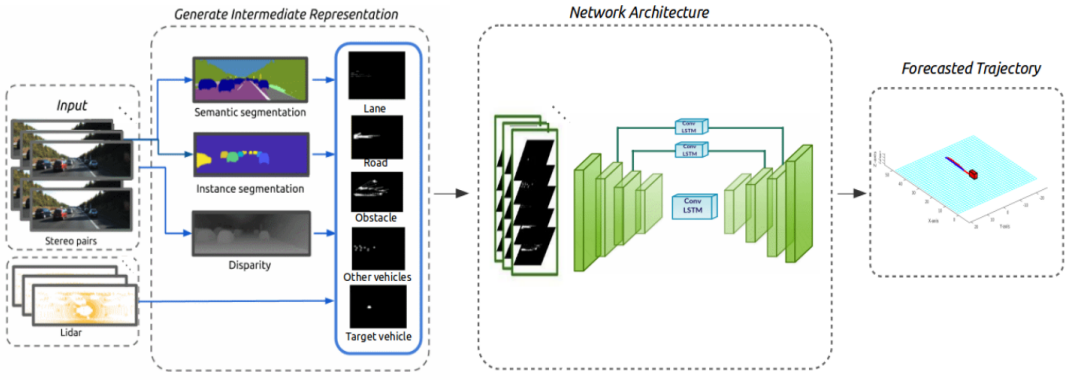

Figure 2.4: INFER network architecture. Input from monocular images and depth information from LiDAR or stereo cameras are used to create an intermediate representation comprising of 5 semantic channels: lane, road, obstacle, other vehicles, target vehicle before being fed to the network (From [37])

.

2.2.3. Encoder-Decoder Network

In the encoder-decoder architectures, an encoder takes in the input sequence and converts it into a fixed-length hidden representation, and the decoder part takes as input the hidden representations and converts it into a required output sequence vector[60].

In [37], the proposed DESIRE network uses an encoder-decoder network architecture to predict the trajectories of an agent in dynamic scenarios. The network consists of 3 parts: an encoder-decoder module to encode the past observations of all the agents in a scene, a Conditional Variational Auto-Encoder (CVAE) to create multiple possible trajectories that account for the uncertainties in a dynamic environment followed by an Inverse Optimal Control based ranking and refinement module as shown in Figure 2.5.

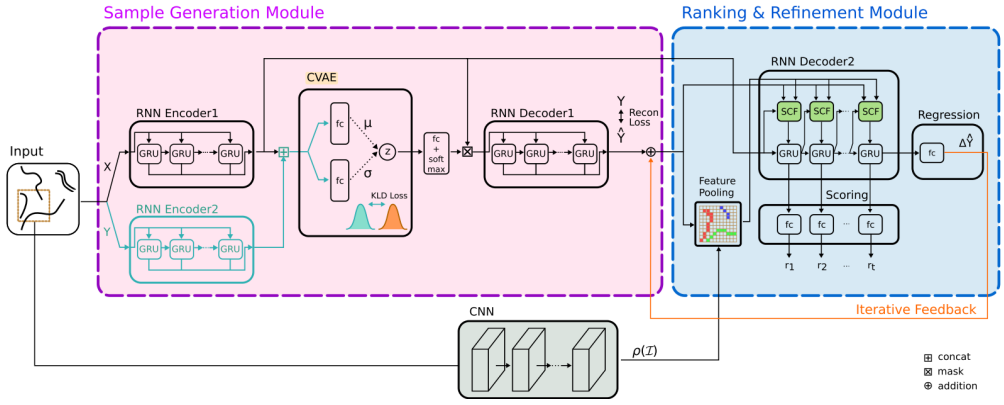

Figure 2.5: DESIRE network architecture. Model is wrapped into a end-to-end trainable encoder-decoder network. CVAE generates multiple plausible predictions and the IOC based module assigns rewards and rank predictions generated by CVAE(From [37])

2.2.4. Attention Network

Recurrent neural networks with recurrent layers such as LSTM and GRU have been the popular approach for sequence modelling problems [60]. However, the sequential nature means that with increased complications, the computational cost increases and this issue is exacerbated in a high dimensional input space [55]. In an encoder-decoder architecture, the encoder summarizes the input into a single context vector which the decoder uses to generate predictions. This technique works well for smaller sequences; however, as the length of the sequence or dimension increases, a single vector becomes a bottleneck and it gets challenging to summarize long sequences or a high dimensional vector space into a single vector. This is where the attention mechanism [62] has distinct advantages. Attention mechanism can help weigh and reason about the relative importance of different features for the prediction task and has shown good performances in the high dimensional input space by being able to identify smaller discriminative regions to focus on[53].

In CAR-NET, proposed in [53], the entire raw image of the scene is supplied as the feature vector. In this high dimensional input space, visual attention module compromising of a single-source and multi-source attention is used to identify and focus on smaller discriminative regions in the image that captures the agent-space interaction. Multi-source attention mechanism (hard-attention) help identifying relevant cues that are scattered over multiple areas in the entire image. In contrast, the single source attention method (soft-attention) draws out essential information from the local area where the agent is situated(Figure 2.6).

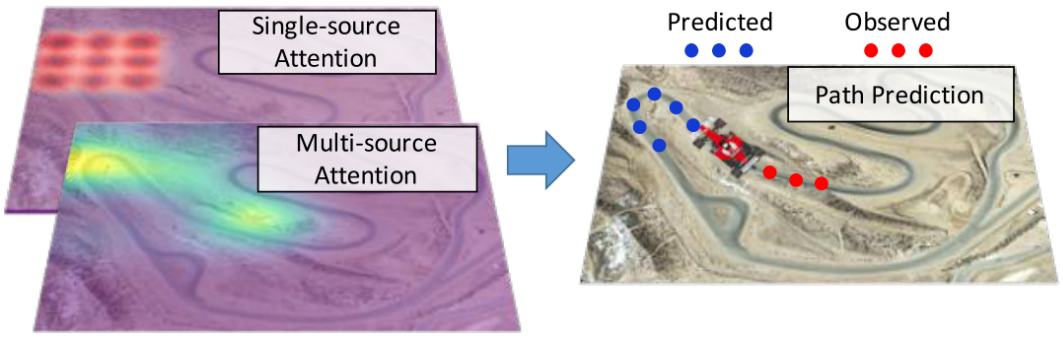

Figure 2.6: Focus regions of a single source and multi-source attention mechanism used in [53]. Single source attention focuses on a single part of the image whereas the multi-source attention weighs the relative importance of each individual pixel

In [48], a joint intention estimation and trajectory prediction framework is utilized as shown in Figure 2.7. The network comprises of 3 separate network units. The last-k sequence of images of a pedestrian is used as the input for intention estimation and trajectory prediction. The third unit predicts the future velocity of the vehicle. The results from the intention estimation network and the last-k sequence of images are first passed through a temporal-attention layer network and then fed as an input to the trajectory estimation network. Temporal-attention layer identifies the relative importance to be given to each observation. However, the attention is not explicitly applied over the pixel-space.

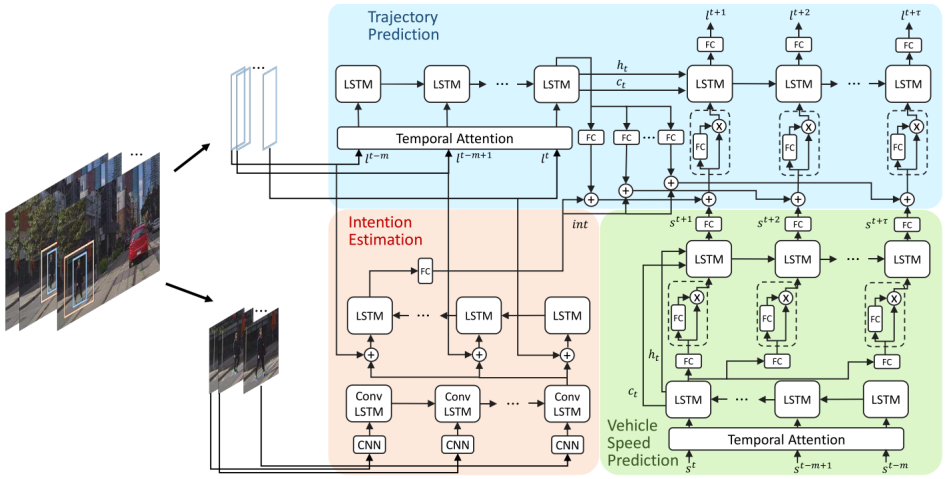

Figure 2.7: Intention Estimation and Trajectory Prediction framework proposed in [48]. The network receives a cropped sequence of images of a pedestrian and the current speed of ego-vehicle. The intention estimation module produces a hidden state representation which is then concatenated with multiple outputs and finally fed to the network. Vehicle speed prediction network use the current location and speed to predict the future speed of the ego-vehicle.

Attention mechanism has also been used to model social trajectory prediction. In [63], proposed 'Social Attention Network' uses the attention mechanism to capture the relative importance of other agents in the scene to predict future trajectories. Multi-Agent Tensor Fusion Network (MATF) proposed in [68] is one of the first networks to include scene context in social trajectory prediction. Encoding of the scene features is combined with a spatially aligned encoding of past trajectories. The spatially aligned encoding of multiple agents is fused through a convolutional mapping that can capture the dynamic interactions of people and the environment as a whole. An attention mechanism is used to identify which agents in the scene will affect the trajectory prediction of a given agent [62]. Uncertainty in the real-world trajectory is captured by generating multiple trajectories using the Conditional GAN [41]. This network is able to model both the spatial constraints and social interactions.

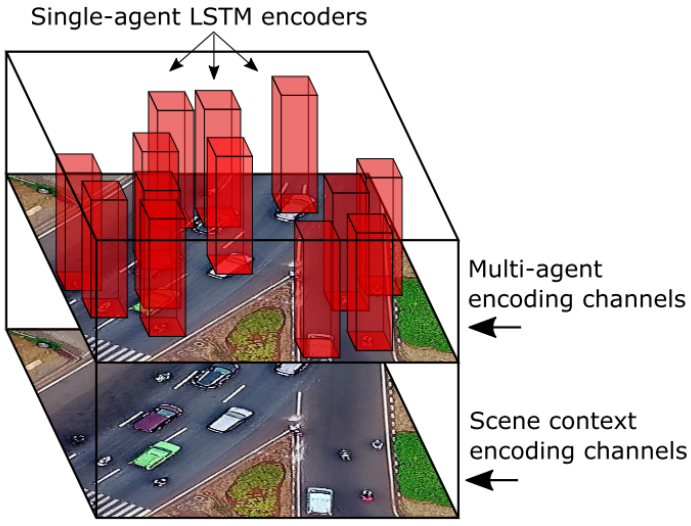

Figure 2.8: Illustration of inputs to the MATF network. Input to the network are past trajectories of 'n' agents fed to LSTM encoders and then the output from agent channels are spatially aligned with the scene context

2.3. Contextual cues

The challenges path prediction of a VRU arises from the fact that there is a sudden change in their motion modes when there is a switch from walking/cycling to abrupt stopping or taking a turn [35]. A human driver can reason about these changes by taking into account the previous behaviour, surrounding environment, rules and regulations, interaction with other agents in the environment, and so on. All this information and signals that influence motion behaviour are termed as contextual cues. Contextual cues can be categorized into three broad categories [52] as shown in fig. 2.9. In this section a more detailed look into the categorization and some of the contextual cues that have been used for path prediction is presented.

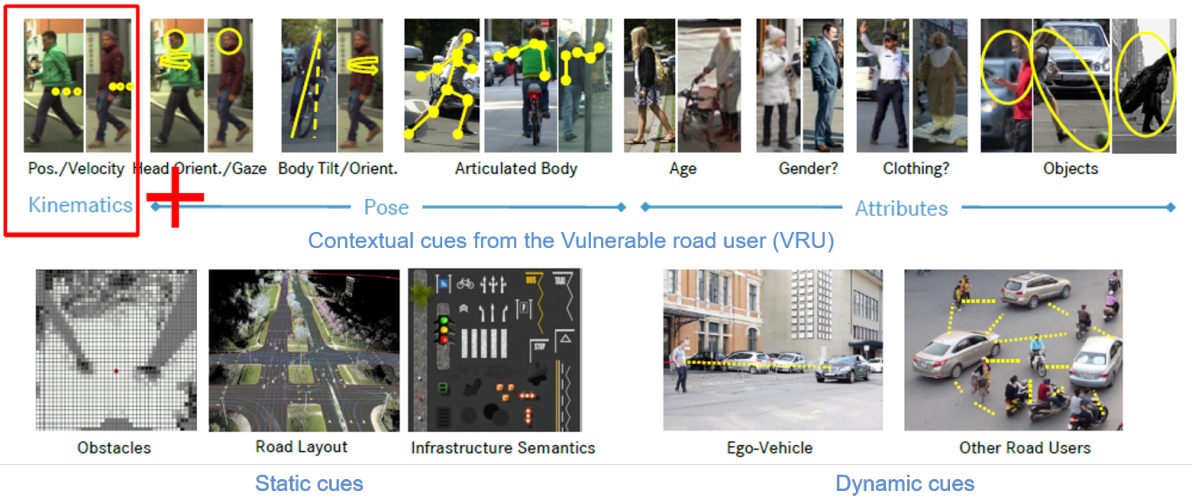

Figure 2.9: Broad categorisation of the contextual cues

2.3.1. Contextual cues from the VRU

Humans usually rely on non-verbal cues to gauge the intentions of other road users, and trajectory prediction methods should be able to account for this information. Moreover, in a naturalistic driving scenario, failure to understand these intentions can lead to accidents. Contextual cues such as head orientation, pose, awareness of the approaching vehicle etc. helps in reasoning and identifying the sudden changes in the motion of a VRU even before they occur [35][45][66]. Also, the abrupt switches in the motion from moving to stopping or vice-versa can be indicated by a VRU through hand gestures or by eye contact and as such contextual cues or signals sent out by the VRU can provide the path prediction models with valuable insight into how the scene will evolve. Including information such as age, clothing and objects carried, allows defining multiple motion types for different groups based on their attention to the environment, reaction time and personality [40][47].

The detection and identification of head orientation have shown to provide a measure of the attention received by various elements in the environment [4]. Head orientation help in the assessment of the situational awareness of the VRU. This knowledge is useful in situations involving the decision making for a VRU, which is affected by the perception and cognition of the approaching vehicle. A glancing look or a VRU monitoring the traffic is a reliable indicator of crossing intentions [47]. Also, an inattentive pedestrian may fail to recognize the approaching vehicle and cross, leading to a hazardous situation. With a measure of situational awareness, a trajectory prediction method can reason whether the pedestrian would stop or not before there is an actual change in the motion dynamics. In [35], including the parameter "Has Seen Vehicle" (HSV) in the DBN along with other contextual cues, improves the log-likelihood of the predictions.

Hand gestures are another contextual cue that a human driver is easily able to identify and is a typical way

that a VRU or other road users interact with human drivers. In [47], it was shown that in more than 90% of the clips, non-verbal communication existed between road users. In [44], including the cue of whether the cyclist's arm was extended or not, improved the prediction task of a cyclist taking a turn. Hand gestures are also used by other road users to acknowledge the awareness and signalling the intention.

Articulated body pose is a more generic contextual cue that can be used to identify intention and awareness of a pedestrian and can potentially capture the head orientation, hand gestures and changes in the gait that can potentially precede any change in the motion [45]. Recent research has shown that the full-body appearance of a pedestrian improves the classification task of crossing/not crossing compared to evaluating only on the head orientation or lower body [46]. However, there are challenges in using this method as robust detection of the pose becomes the bottleneck. Occlusions are quite common in a naturalistic dataset with the VRU occluded by objects or other agents. Also, the evaluation of complete body posture may require stereo cameras or motion history of images.

Optical flow is another generic cue that provides full-body information. Optical flow is the estimation of apparent velocity of an object measured by the change in brightness over the subsequent sequence of images. Relative motion between the observer and the object causes the change in brightness at the pixel level in images [26]. Research in the latter part of 1970's on optical flow has shown that it can be used to segment images into different objects [30], to extract the shape of the object[14] and in recovering the relative velocity of the object. All these inherent attributes make optical flow an interesting and attractive contextual feature for deep learning model with a large training set data. Optical flow may implicitly detect changes in body postures, speed changes, signalling etc. In [32], the histogram of optical flow features was used to capture the motion differences between the upper torso and legs of a pedestrian and used as a contextual cue to predict future trajectories. In [58], the dense optical flow was directly used as a feature for the dynamic trajectory predictor.

2.3.2. Static Cues

The surrounding environment and agents affect the motion of a VRU. All these contextual cues originating from the surrounding infrastructure, traffic rules, street layout and other agents are classified as static cues.

The agent's behaviour and motion will be influenced by their position and layout of the environment [43]. For example, when a pedestrian approaches a curbside, there would be a higher probability of stopping and similarly an obstacle in the path may lead to the agent changing directions. Also, the layout has a direct correlation with the amount of attention a VRU pays to the environment. In [47], it was found that on a non-designated crosswalk or a narrow street, the pedestrian pays more attention compared to a designated crosswalk such as zebra crossing. In [34], Kitani et al. showed the impact of physical scene features such as destinations, road and sidewalks. In [31], it was seen that the knowledge of the destination yields better recognition of human activity.

Static cues can be incorporated into the model through multiple ways: map-aware methods can use a pre-loaded map of the environment, sensors can detect obstacles and dynamic objects, or a semantically rich representation of the environment can be created and used as static cues [37].

2.3.3. Dynamic Cues

Dynamic cues relate to interaction and awareness of the VRU with respect to the other agents in the environment. Under these, we have position and awareness of the approaching ego vehicle, the effects of other road users and social interactions. In the survey conducted in [47], it was seen that pedestrians intention to cross the road is affected by the gap between the agent and the ego-vehicle or more precisely Time to Collision (TTC). Apart from this, social grouping, formations, and group movement are other aspects that fall

under dynamic cues [42] [2]. Identifying formations and convoys allow incorporating motion models that essentially differs from a model of an isolated individual agent.

2.4. Datasets

Dataset and benchmarks play an important role in the development and improvement in the accuracy of deep learning models [16]. Benchmarking helps in measuring and comparing the performances, assists the improvement of the state-of-art in their respective domains. For trajectory prediction, there are multiple varieties of datasets. These vary in terms of view-point, type of annotations and sensors. "Stanford Drone Dataset" [50], "ETH" [18] and "inD Dataset" [7] are examples of the top-down view datasets recorded from a drone or surveillance cameras perspective containing world coordinate information. Then there are datasets recorded by a camera mounted on-board a moving vehicle in naturalistic driving scenarios: Pedestrian Intention Estimation (PIE) [48], Joint attention in Autonomous Driving (JAAD) [36], EuroCity Persons (ECP) [9] , Argoverse [11] , Waymo [1], nuScenes [10], Lyft Level 5 [33] and KITTI[20]. Among these, KITTI, Waymo, Argoverse, nuScenes and ECP dataset contains the ego-motion and 3-D coordinates of a VRU along with camera recordings. Unlike ECP-Dense dataset, KITTI, Argoverse, Waymo and nuScenes do not contain separate curated tracks of VRU for path prediction. In section 4.1, a detailed look into the ECP-Dense dataset is given. For cyclist and other riders, the Tsinghua-Daimler Cyclist (TDC) [38] dataset has been used as a benchmark. Further details and comparison based on sensor types, number of pedestrians and riders and whether for privacy reason faces are blurred or not is made in section 2.4.

			Annotations (unique)			
Dataset	View	Sensors	Pedestrians	Riders	Freq	Faces blurred
KITTI[20]	Vehicle	LiDAR (10 Hz) + camera (10 Hz) + GPS/IMU	9400	~ 3300		No
EuroCity Persons[9]	Vehicle	camera (20 Hz)	218313	19780	20 Hz	No
Waymo open dataset[1]	Vehicle	5 LiDAR + 5 cameras + GPS/IMU (all 10 Hz)	2759030 (22902)	66643 (620)	10 Hz	Yes
nuScenes [10]	Vehicle	1 x LiDAR (20 Hz) + 5 x radar (13 Hz) + 6 x camera (12 Hz) + GPS/IMU	222164		2 Hz	Yes
Argoverse [11]	Vehicle	LiDAR (10Hz) + 7 x camera (30hz) + stereo (5Hz) + maps	78080 (867)	10623 (84)	10 Hz	Yes
Lyft Level 5[33]	Vehicle	3 x LiDAR (10Hz) + 7 x camera (30 Hz) + maps	24395		10 Hz	Yes
City Persons [67]	Vehicle	Camera	31514	3502	1.5 Hz	No
PIE [48]	Vehicle	Camera	740000(1800)		30Hz	No
ETH [18]	Top-view	Camera	8908		2.5 Hz	No
Stanford drone dataset [50]	Top-view	Camera	11216	6364	25 Hz	No

Table 2.1: Overview of datasets by comparing the type of sensor information available, no. of annotations for VRU and the frequency at which they are annotated.

2.5. Comparison of performance

For trajectory prediction of a pedestrian in the image plane, JAAD and PIE dataset serve as a good benchmark with curated tracks of pedestrians performing interesting motions. The error in the benchmarks are reported in terms of Average Displacement Error (ADE) and Final Displacement Error (FDE). ADE measures the dissimilarity by taking the mean of the euclidean distance between the predicted points and the ground truth over the entire $t+n$ prediction steps. Whereas, FDE measures the distance between the final predicted point in the prediction step and the corresponding ground truth position.

The ADE of methods on the JAAD and PIE dataset is reported in table 2.2.

Method	ADE on PIE			ADE on JAAD		
	0.5s	1s	1.5s	0.5s	1s	1.5s
KF	123	477	1365	223	857	2303
LSTM	172	330	911	289	569	1558
Bayesian-LSTM [5]	101	296	855	159	539	1535
PIE_{traj} [48]	58	200	636	110	399	1248

Table 2.2: ADE of pixel bounding boxes predicted over all time steps on PIE and JAAD dataset [48]

Stanford drone dataset [50] has been used as the benchmark for trajectory prediction of pedestrians in world coordinates. The performance of various methods in terms of Average Displacement Error and Final Displacement Error for 4.8s prediction horizon using 3.2s of motion history is listed in the Table 2.3. The best performing methods on the reported results are RNN based. The comparison shows that "Sophie" [54] which is a CNN based attention network has state-of-art result closely followed by "Desire" an encoder-decoder type architecture.

Method	ADE in cm (4.2s)	FDE in cm(4.2s)
LSTM	37.35	77.13
Social Force [24]	36.38	58.14
Social LSTM [3]	31.19	56.97
Social GAN [22]	27.25	41.44
Desire[37]	19.25	34.05
CAR-Net[53]	25.72	51.8
Sophie [54]	16.27	29.38
MATF GAN [68]	22.59	33.53

Table 2.3: Average Displacement Error and Final Displacement Error over 4.8s prediction horizon in Stanford Drone Dataset. [68]

2.6. Contribution

Contextual cues originating from a VRU can indicate the awareness of the environment, intentions, and also as a feature used for communication of those intentions with other agents in the environment. Taking all these cues into account by creating a separate detector for each is an impossible task. Instead, looking at a generic VRU related feature which can be used in an end-to-end training would be ideal. From a generic contextual cue perspective, articulated body pose and optical flow are certainly of interest. Articulated body pose still has issues with robust detection and identification in a naturalistic driving scenario. Whereas, optical flow is an interesting generic contextual cue with implicit property to capture the relative motion between body parts [32] that a non-linear regression approach can use to capture the signs preceding any change in motion of a VRU. However, the dense optical flow is a high-dimensional input compared to a 2-d position

feature (x, y) that is the basic input for the path prediction task. So, experiments would be conducted to evaluate the benefits of using a high-dimensional dense flow with spatial relations compared to processing it to a lower-dimensional feature.

The physics-based approaches require explicit modelling of the situations but are useful in providing the reasoning behind predictions [35]. This is interesting from the perspective of getting a greater insight into the trajectory prediction problem but is limited by the vast number of cases in a naturalistic driving scenario. Among pattern-based approaches, clustering-based approaches such as PHTM that requires matching from the database would become slower with larger datasets. In the literature and benchmarks (section 2.5) a shift towards a non-linear regression approach utilizing RNN and other deep learning methods can be seen and they have consistently improved the prediction performances. Attention mechanism has been used to find important regions in a high-dimensional space [48] [55] [62], LSTM and GRU have been the preferred backbone to model the sequential nature of predictions [68] [37] [56]. In some of the cases utilizing images to extract cues, Convolutional Neural Network (CNN) has been the preferred feature extractor as it can identify geometrical shapes and spatial relations. Even GAN have been used to generate multiple plausible trajectories to capture the uncertainties of a real-world scenario. For this work: a non-linear regression approach utilising multiple methods such as RNN to model the sequential nature of the track, CNN to extract features from the dense optical flow and attention mechanism to weigh the relevant regions in dense optical flow will be combined to design the network architecture.

The main inspiration for the attention mechanism used in this work is from [65] which introduced visual attention for image captioning. In this work, 2 types of attention mechanisms were utilized: soft-attention and hard-attention. Soft-attention assigns weight to the entire input, higher weights are indicative that it is being attended to more. Whereas, hard attention focuses only on the subset of the input, in the case of images this results in the network seeing a small extracted region of the image, which, hopefully is the part most relevant to the task. Since for this work the dense flow of the bounding box of the relevant pedestrian is only fed as input, soft-attention will be useful to identify the relevant pixels within that region as the entire scene is not fed as the input to call for adding hard-attention.

Among the naturalistic driving dataset presented in section 2.4, PIE [48] and JAAD [36] lacks the world-coordinate information. Argoverse [11] , Waymo [1], nuScenes [10] and Lyft Level 5 [33] have the faces of pedestrians blurred which would introduce an artifact to optical flow features. ECP-Dense dataset which is a curated subset of interesting scenarios for path prediction from the much larger ECP dataset also contains ego-motion and 3-D coordinates of a VRU along with camera recordings. Hence, ECP-Dense dataset is chosen to evaluate the performance of models presented in this work. Apart from this, further processing and labelling of the tracks in ECP-Dense is done, details of which are presented in Section 4.1.

3

Methodology

The goal of this work is to investigate the incorporation of a generic contextual cue in optical flow into a Recurrent Neural Network. This chapter is organised such that, in section 3.1, the extraction and processing of optical flow features is presented. In section 3.2, RNN architectures used to incorporate the optical flow feature in this work is explained. Section 3.2 starts with the comparison of the fundamental recurrent units GRU and LSTM and selection of one of these recurrent units as the building block for the networks. Then the network architectures used to incorporate optical flow features and the soft-attention mechanism used to handle the high dimensionality of the features is elaborated in section 3.3. In section 3.4, the method used to visualize the attention weights or focus areas of where exactly the network is looking in the denseflow features to make predictions is presented. The coordinate transformation required to combine the contextual cue which is available in the ego-vehicle coordinate system and the 2D world position of a pedestrian is presented in section 3.5. Finally, section 3.6 gives details about the evaluation metrics to compare the performance.

3.1. Optical flow as contextual cue

Optical flow is not directly measurable and requires precise correspondence of features between 2 images and as such has been a difficult task to solve in the field of computer vision. Traditional methods involve complex optimization steps [59], making the estimation of optical flow slow.

However, with deep-learning, optical flow can be learned using the end-to-end learning approach and CNN has the ability to learn and correlate features irrespective of the spatial position. The advantage of using a CNN is that it is comparatively faster than the traditional approaches and can run in real-time. In this field, FlowNet [17] was one of the first end-to-end CNN architecture for optical flow estimation and was able to make predictions at the rate of 10fps. FlowNet network was a milestone in terms of displaying the capability of learning-based approaches to estimate optical flow. However, the performance was not competitive with the existing state-of-art traditional methods. The successor, FlowNet 2.0 [28] was designed to address the issue of performance. It is a cascaded network that refines the flow estimate from the preceding step and also includes a part that specializes in detecting and estimating small displacements. FlowNet 2.0 was marginally slower than its predecessor but decreased the flow estimation error by 50%. Also, the network had shown comparable performance with respect to the state-of-art methods while being able to make predictions in real-time.

For extracting optical flow features, the LiteFlowNet network proposed in [27] will be used. The choice of this network is inspired from the fact that LiteFlowNet [27] is an alternative to FlowNet 2.0, it is lightweight (model is 30 times smaller than FlowNet 2.0), fast (1.36 times faster) and accurate CNN based optical flow estimation method. It is composed of 2 sub-networks (NetC and NetE) as shown in fig. 3.1 specializing in pyramidal feature extraction and optical flow estimation, respectively. NetC consists of a two-stream network to extract feature descriptors from the image pair, and NetE consists of specialized modules for cascaded flow inference and regularisation.

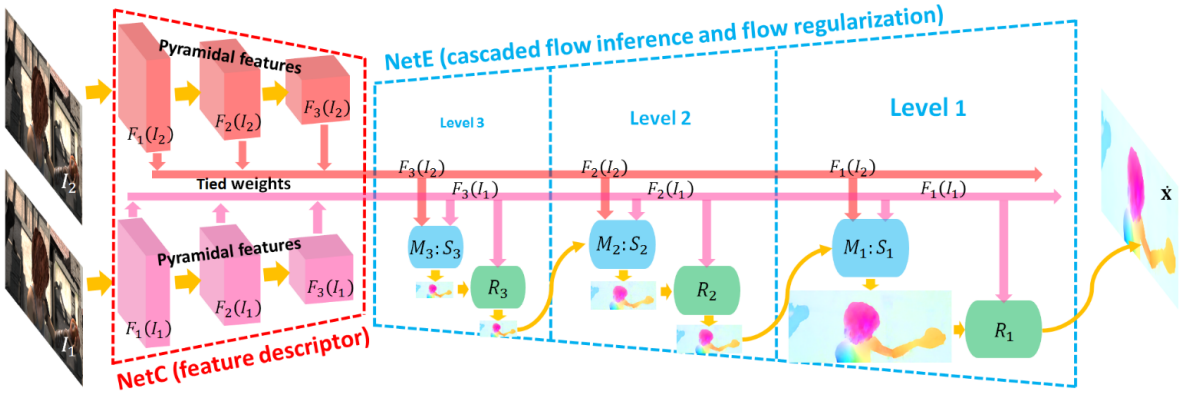

Figure 3.1: The network architecture of LiteFlowNet. NetC extracts high-level pyramidal features from a pair of images and NetE does a cascaded flow inference from coarse to fine using the pyramidal features and flow estimation from the previous level (from [27])

3.1.1. Processing pipeline for extraction of features

The pipeline for extraction and processing of optical flow features used in this work is as shown in fig. 3.2. Optical flow will be estimated by passing the entire image of the scene at t and $t-1$s through LiteFlowNet network. The network will output pixel-wise denseflow features ($V = [V_x, V_y]$) which shows the relative movement of a feature from one frame to the other in horizontal and vertical direction respectively. The flow features belonging to the pedestrian is extracted using the ground truth bounding box position of the pedestrian. However, the bounding box will still contain some background pixels not belonging to the pedestrian body.

Optical flow features located on the pedestrian body is extracted by generating a mask of the pedestrian pixel using semantic scene segmentation. Semantic scene segmentation is the task of classifying each pixel in an image to a different class of object. DeepLabv3[12] network is selected for the segmentation task as it has comparable performance to state-of-art results in PASCAL VOC 2012 semantic image segmentation benchmark [19]. Also, the code-base of DeepLabv3 is open-sourced, enabling easy reproduction of the processing pipeline. For a given ground truth bounding box position of the pedestrian, the flow values of the pixels not belonging to the pedestrian class is set to zero.

The flow values are normalised by the distance of the pedestrian from the ego-vehicle and also by removing the median motion of pedestrian pixels. An object closer to the camera will have larger pixel displacement and flow for the same motion compared to an object far-away from the vehicle. As shown in eq. (3.1), the horizontal and vertical component of flow is normalised by the distance to the ego-vehicle ($dist$) available from the LiDAR point cloud and a constant(c). For pedestrian motion compensation, the median pedestrian flow is estimated by using the pedestrian mask at the pixel level and subtracted from the flow values belonging to the pedestrian body.

$$v = \frac{V.dist}{c} \tag{3.1}$$

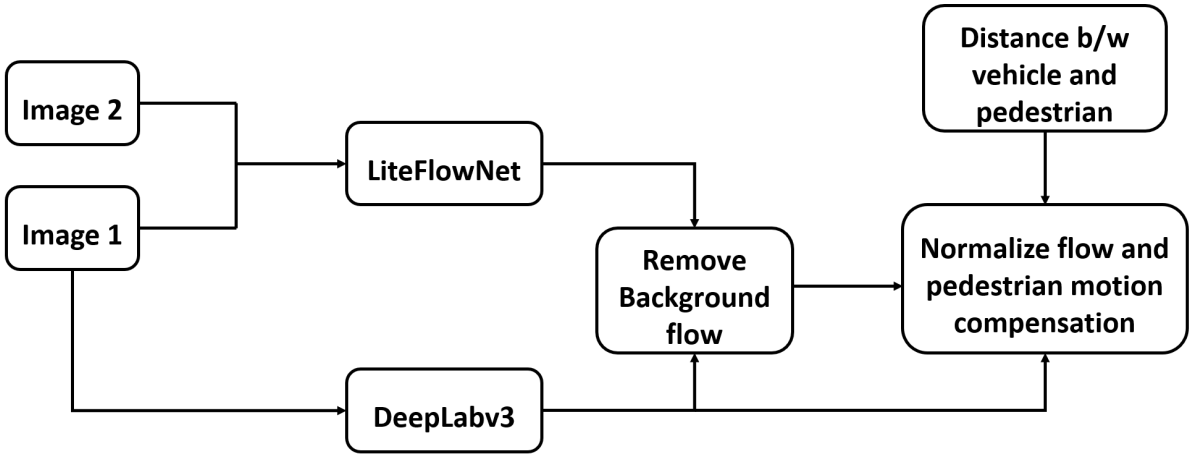

Figure 3.2: The processing pipeline to estimate optical flow and extraction of features. The optical flow is estimated using the liteFlowNet, the background is removed by using the segmentation network DeepLabv3 and is also used to calculate the median flow from the pedestrian pixels in the bbox. The optical values is normalised based on the distance of pedestrian from the ego-vehicle as closer the pedestrian higher is the estimate optical flow values

3.1.2. Flow feature derivatives

The extracted and processed denseflow features can be incorporated into an RNN architecture in multiple ways. Direct pixel-wise flow features can be fed to the network or dimensionality reduction can be done at the cost of loss of information or spatial relations. In the sub-sections below, the optical flow features starting with the full pixel-wise flow to the histograms of flow features used for experimentation is explained.

Denseflow features

The original optical flow in the bounding box of a pedestrian is resized to a fixed feature vector of size $64 \times 32 \times 2$ ($height \times width \times channels$) using interpolation. Then, the normalised denseflow features ($v = [v_x, v_y]$) are used to calculate the orientation $\theta = atan2(v_y, v_x)$ and $magnitude = \sqrt{v_x^2 + v_y^2}$ for each pixel in the resized bounding box. These features would be denoted by 'Denseflow' in the upcoming sections and experimentation's.

Histogram of body

Histogram of flow features is calculated on the optical flow in the original bounding box by assigning the weighted contributions of the magnitude of denseflow features($magnitude = \sqrt{v_x^2 + v_y^2}$) to bins b $\in [0, 15]$. The contributing bin is evaluated using the orientation $\theta = atan2(v_y, v_x)$ and is assigned to bins belonging to the index $b \in \left\lfloor \frac{\theta}{\pi/8} \right\rfloor$ as shown for one of the sample scenario in fig. 3.3. Bin contributions are normalized by the total number of pixels with pedestrian mask in the bounding box contributing to the histogram. To capture the motion differences between upper and lower body, the bounding box is split into 2 [32] and the resulting histogram features are concatenated. This results in a feature vector of size 32 indicated by the term ' Hist body'.

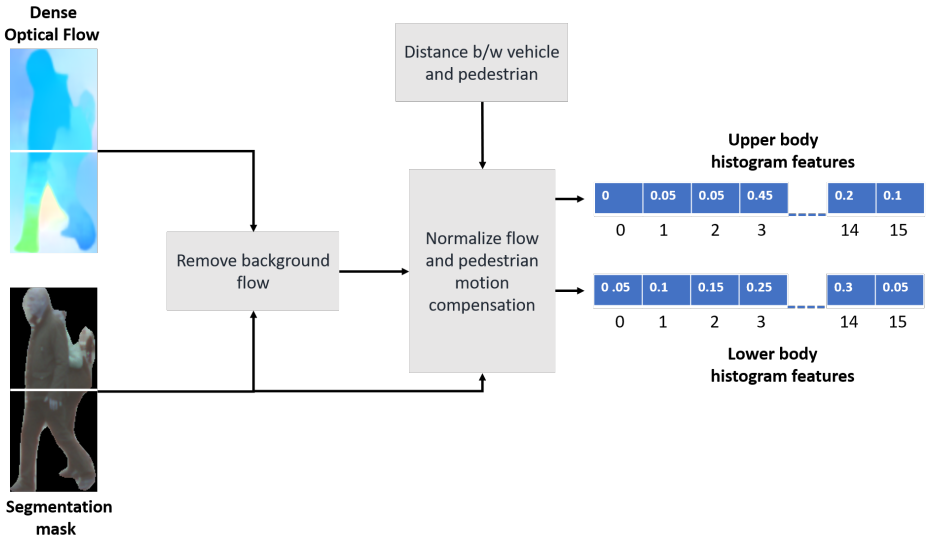

Figure 3.3: Calculating hist of body for a sample case. The processing pipeline takes in the segmentation mask and dense optical flow features. The histogram features are calculated by assigning the weighted contribution of flow magnitude to bins corresponding to flow orientation calculated at each pixel location with pedestrian mask.

Histogram Cell

Histogram of gradients used for pedestrian detection in images was introduced in [39] and [15]. Following the above principles, the histogram features of dense flow would be estimated. Initially, the bounding box of a pedestrian is resized to $64 \times 32 \times 2$. After this step, the flow bounding box is divided into 8×8 cells with stride 1 (resulting in 32 cells) and histogram of flow features is calculated for each of the cell by using the 360 degree of orientation and assigning it according to bin index $b \in \left\lfloor \frac{\theta}{\pi/8} \right\rfloor$ as mentioned above. After this, normalisation is done block wise (block is a group of cell). A 2×2 cell blocks are created with a stride of 1 along both direction resulting in 7×3 blocks. Normalisation is done by accumulating the histogram energy (L1 norm) in each block and is used to normalize each cell in the block. Individual cells is shared between several blocks but its normalisation is block dependent and thus different. This results in a feature descriptor of size $7 \times 3 \times 2 \times 2 \times 8$. This histogram of features would be described as 'Hist cell' in the coming experiments.

3.2. Recurrent Neural Network

RNN's have shown promising results in many of the time-series and sequence prediction tasks such as machine translation, music generation, path prediction and market predictions [60] [8]. An RNN is a type of deep learning method that can handle variable-length input sequences and learn the underlying pattern of a time-series or sequence. Recent successes of RNN can be attributed to the sophisticated recurrent units such as GRU [13] and LSTM [25]. A detailed comparison is given below, looking in depth how both units differ from each other and their advantages.

A LSTM unit [21] as shown in fig. 3.4, takes in $x_t \in \mathbb{R}^d$ and the previous hidden state (h_{t-1}) and cell state (C_{t-1}) as input at time t. Cell state contains the intermediate results from one iteration step to another and the hidden state provides a snapshot of the step's result. An LSTM unit is made up of three sigmoid gates (forget, update and output) indicated by σ in fig. 3.4. Each gates perform a specialised task.

Forget gate helps in erasing a certain amount of information from the previous observations (eq. (3.2)) and is given by:

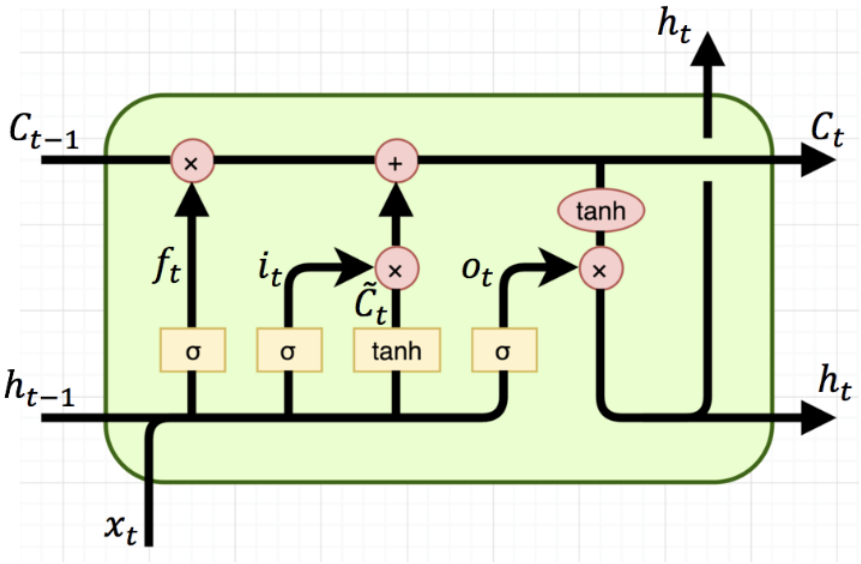

Figure 3.4: A LSTM unit that takes in input vector x_t and provides the hidden state for next iteration by keeping the relevant information from current and previous time steps.[1]

$$f_t = \sigma\Big(x_t W_f + h_{t-1} U_f + b_f\Big) \tag{3.2}$$

Where $W_f \in \mathbb{R}^{h\times d}$, $U_f \in \mathbb{R}^{h\times d}$ and $b_f \in \mathbb{R}^{h\times d}$ are the trainable weight and bias parameters of the forget sigmoid gate that is learned during training. h is the hidden dimension and d the input vectors dimension.

The update gate vector (i_t) retains a certain amount of current information that will be updated to the cell activation state. It takes in the current input (x_t) and the previous hidden state (h_{t-1}) as input to the sigmoid gate with trainable weight and bias parameters given by W_i, U_i and b_i.

$$i_t = \sigma\Big(x_t W_i + h_{t-1} U_i + b_i\Big) \tag{3.3}$$

Finally, the output gate computes the usage of information for the next iteration and stores it in o_t. The trainable weight and bias parameters is given by W_o, U_o and b_o.

$$o_t = \sigma\Big(x_t W_o + h_{t-1} U_o + b_o\Big) \tag{3.4}$$

The information from the forget and update gate is combined to compute the cell input activation vector ($\tilde{C}_t$) that is read from and written to during each time step. Next, the previous cell state and the current cell activation vector is combined to give the cell state (C_t) for current time-step.

$$\tilde{C}_t = tanh\Big(x_t W_g + h_{t-1} U_g\Big) \tag{3.5}$$

$$C_t = \sigma\Big(f_t C_{t-1} + i_t \tilde{C}^t\Big) \tag{3.6}$$

The hidden state is evaluated as shown in eq. (3.7) by taking as input the cell state and output gate vector which contains the usable information.

$$h_t = tanh(C_t) * o_t \tag{3.7}$$

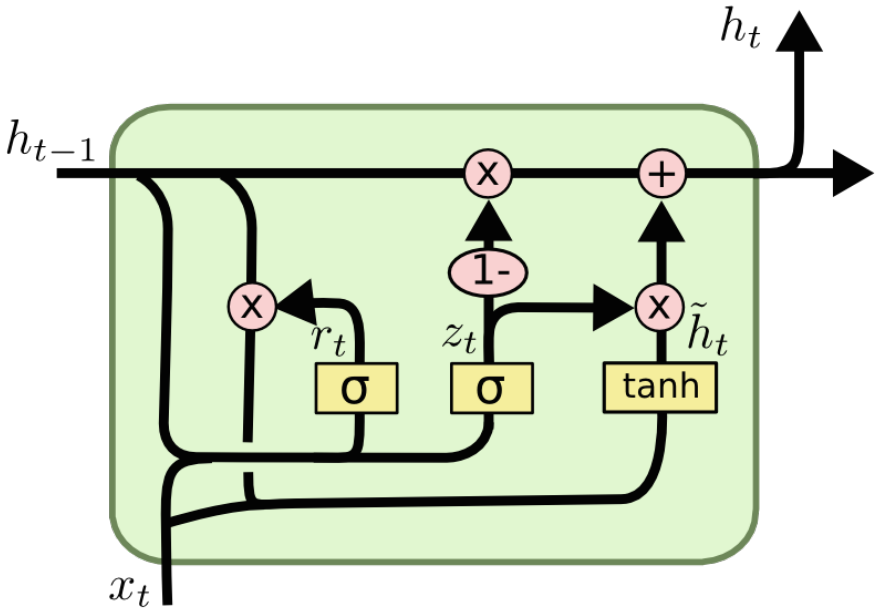

Figure 3.5: A GRU unit showing the combination of input and forget gates into a single entity unlike LSTM. [2]

Gated Recurrent Unit is an alternative to LSTM that has been developed as a lighter unit with fewer parameters enabling faster training. GRU has only two sigmoid gates: reset gate (eq. (3.8)) and update gate (eq. (3.9)). The forget and output gate of the LSTM (eq. (3.2) and eq. (3.4)) is combined into a single reset gate in GRU (eq. (3.8)).

$$r_t = \sigma\Big(x_t W_r + h_{t-1} U_r + b_r\Big) \tag{3.8}$$

$$z_t = \sigma\Big(x_t W_z + h_{t-1} U_z + b_z\Big) \tag{3.9}$$

Also, there is no separate cell state and hidden state as in LSTM (eq. (3.6) and eq. (3.7)). Cell state adopts the functionality of the hidden state in GRU as shown in fig. 3.5.

$$\tilde{h}_t = tanh\Big(x_t W_h + r * U_h . h_{t-1} + b_h\Big) \tag{3.10}$$

$$h_t = z_t * h_{t-1} + (1 - z_t) * \tilde{h}_t \tag{3.11}$$

Where z_t is the update gate vector, r_t is the reset gate vector and similar to the LSTM, W_h, U_h and b_h are trainable weight matrices and bias vectors.

The LSTM and GRU units allow retaining the vital information over a long range of the sequence in their hidden state. GRU unit is a simpler design with fewer parameters than an LSTM unit enabling faster training and lesser memory requirements. LSTM, with its higher parameters, can enable greater information expressibility. However, empirical comparison of the LSTM and the GRU have shown that the type of gated recurrent unit depends on the dataset and corresponding task [13]. An empirical evaluation on the choice of the recurrent unit to be used is made in section 4.3.1. Based on the experimental results, the basic recurrent units for the network architecture is selected.

[1] https://colah.github.io/posts/2015-08-Understanding-LSTMs/
[2] https://medium.com/@saurabh.rathor092/simple-rnn-vs-gru-vs-lstm-difference-lies-in-more-flexible-control-5f33e07b1e57

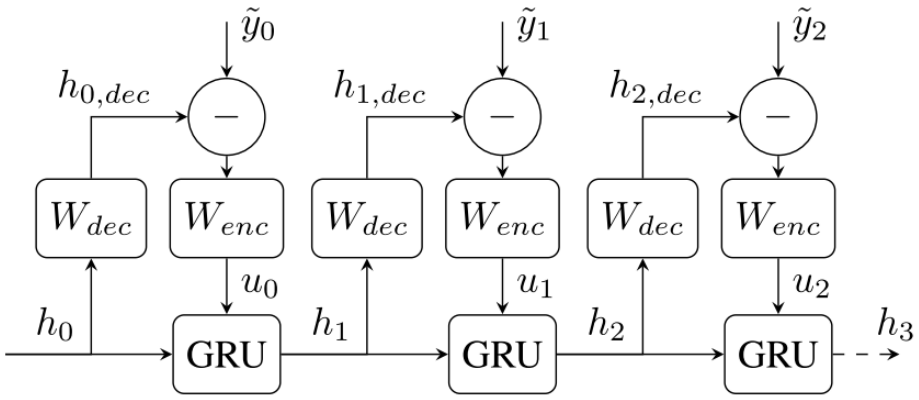

Figure 3.6: The figure shows the fundamental RNN architecture proposed in [44] that the current work builds upon. The figure shows how the positional measurement of a pedestrian is processed over time.

3.3. Network Architectures

In this section, the 3 network architectures used for experimentation and incorporation of position and optical flow features is presented. In section 3.3.1, the baseline architecture from [44] that will be used to take only position as input is elaborated. Building upon this baseline network, in section 3.3.2 soft attention mechanism is introduced to handle contextual cue in the form of histogram of optical flow. Section 3.3.3 introduces a network that can handle the dense optical flow and also keep the spatial relations by using convolutional layers and finally the visual soft-attention to identify focus regions in the spatial layout of the denseflow.

3.3.1. Baseline RNN (Base_RNN)

The work in this thesis builds upon the network proposed in [44] to incorporate the positional features and for predicting the probability distribution at each future time step. The RNN network from [44], uses all the previous measurements $y_{0:t}$ to predict a Gaussian distribution of confidence at future time steps.

The network proposed in [44] is divided into 2 steps: the inference part (fig. 3.6) processes the measurement input over time and the prediction part (fig. 3.7) that shares the encoder, decoder and GRU layers with the inference part to predict n steps ahead. The encoder is a simple linear layer that takes in as the input the difference in position between 2 time steps.

The relative difference in position ($\tilde{y}_t = [x_t - x_{t-1}]$) is fed to the encoder($W_{enc}$) (eq. (3.12)) which is a linear layer with 64 elements, the decoder(W_{pos}) is also a linear layer with 64 elements that takes as input the hidden state of the GRU and predicts the relative position and confidence in terms of Gaussian distribution around it for the future time step. At the start of the track at t_0, the relative difference in position is taken as zero. The input data ($\tilde{y}_t$) fed to the network is normalized by scaling it with the mean and variance calculated over the training data.

$$u_t = W_{enc}\big([x_t - x - t - 1] - [W_{pos}(h_t)]\big) \tag{3.12}$$

In the prediction step, zeros is fed as the relative position. The input signal for time steps $t+1$ to $t+n$ is given by:

$$u_{t+i} = W_{enc}(0) \tag{3.13}$$

The future position and confidence is predicted as shown in fig. 3.7. The predicted position (x_{t+n}) and

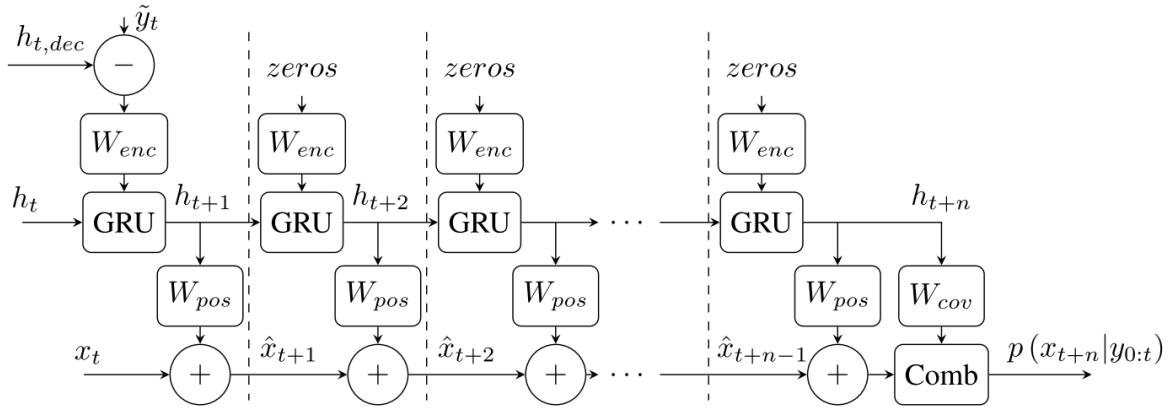

Figure 3.7: An overview of how the RNN predicts future position along with the confidence([44]).

covariance (Σ_{t+n}) for n steps ahead is computed as follows:

$$\hat{x}_{t+n} = x_t + \sum_{i=1}^{n} W_{pos}\left(h_{t+i}\right) \tag{3.14}$$

$$[l^{[0]}\ l^{[1]}\ l^{[2]}]^T = W_{conv}\left(h_{t+n}\right) \tag{3.15}$$

Where W_{conv} is a linear layer in the prediction part of the network. As shown in fig. 3.7, it takes the hidden state of GRU as input and helps in predicting the 2D covariances given by eq. (3.16) - eq. (3.19).

$$\sigma_0 = exp\left(l^{[0]}\right) \tag{3.16}$$

$$\sigma_1 = exp\left(l^{[1]}\right) \tag{3.17}$$

$$\rho = tanh\left(l^{[2]}\right) \tag{3.18}$$

$$\Sigma_{t+n} = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix} \tag{3.19}$$

3.3.2. Attention network (RNN_attn)

Building upon the network proposed in [44], the contextual cue in the form of histogram of optical flow features ('Hist body' and 'Hist cell' indicated by $feat_t$ in fig. 3.8) is passed through an encoder($W_{enc,feat}$) to convert it into a fixed-length hidden representation. The encoders is made up of fully connected linear layer of 256 units. The input positional and flow features are passed through separate encoders as shown in Figure 3.8. In this architecture, the encoder summarizes the input feature into a single context vector. However, as the input dimension increases, a single vector becomes a bottleneck. This is where the attention mechanism has distinct advantages, it can weight and reason about the relative importance of different features for the prediction task and have shown good performances in the high dimensional input space by being able to identify smaller discriminative regions to focus on.

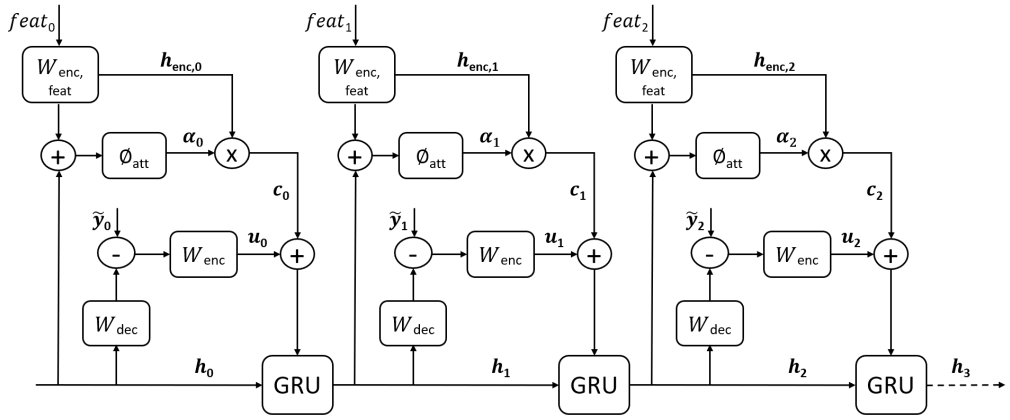

Figure 3.8: RNN attn: The flow features and positional features are encoded by passing through a fully connected layer and the encoded flow is weighted by soft attention before concatenating and passing through a GRU. In the aboove figure '+' indicates the concatenation of 2 vectors and '×' indicates the element wise multiplication of vectors.

The network makes use of soft-attention, the encoded input contextual feature is weighed based on the weights calculated using the previous hidden state from the GRU and the current encoded input context feature. The output of the encoded position and the weighted contribution of encoded features (c_t) is concatenated into a single vector before passing it through GRU to enable prediction.

$$\alpha_t = softmax\Big(\big(\phi_{att}(h_{enc,t}, h_{t-1})\big)\Big) \tag{3.20}$$

$$= softmax\left(W_s \begin{bmatrix} h_{enc,t} \\ h_{t-1} \end{bmatrix} + b_s\right) \tag{3.21}$$

$$c_t = h_{enc,t} \,.\, \alpha_t \tag{3.22}$$

Where c_t is the output from element-wise multiplication of weights (α_t) with the encoded features ($h_{enc,t}$).

The weight matrix α_t is calculated by passing the concatenated hidden state and encoded feature vector through a fully connected layer ϕ_{att} with weight W_s and bias b_s followed by a softmax activation function [53].

Also, the prediction network network takes in as input the predicted optical flow features as input for contextual features [32]. The prediction of optical flow features is done by $W_{pred,feat}$ which is a fully connected linear layer taking as input the hidden state of the GRU (fig. 3.9). Since the GRU takes in as input the encoded past measurement of position and optical flow features, the hidden state of the GRU is used to predict how the flow features would look like in the future time steps.

3.3.3. Convolutional attention RNN (Conv_attn)

Convolutional RNN uses convolutional layers as a feature extractor for images which is then fed to the RNN layers for temporal predictions. Convolutional layers are good at maintaining a spatial invariance (i.e. they are not sensitive to the position of features in a pixel). Since Denseflow (section 3.1.2) gives pixel-wise flow characteristics, convolutional layers will be used to extract spatially invariant features to be then fed to the GRU. As shown in fig. 3.10, the denseflow features are passed through convolutional block consisting of 2 convolutional layers followed by a soft-attention on the the final output of CNN layer before feeding the extracted features through GRU for temporal predictions.

To keep the spatial correspondence between the denseflow features and the final output of the CNN fea-

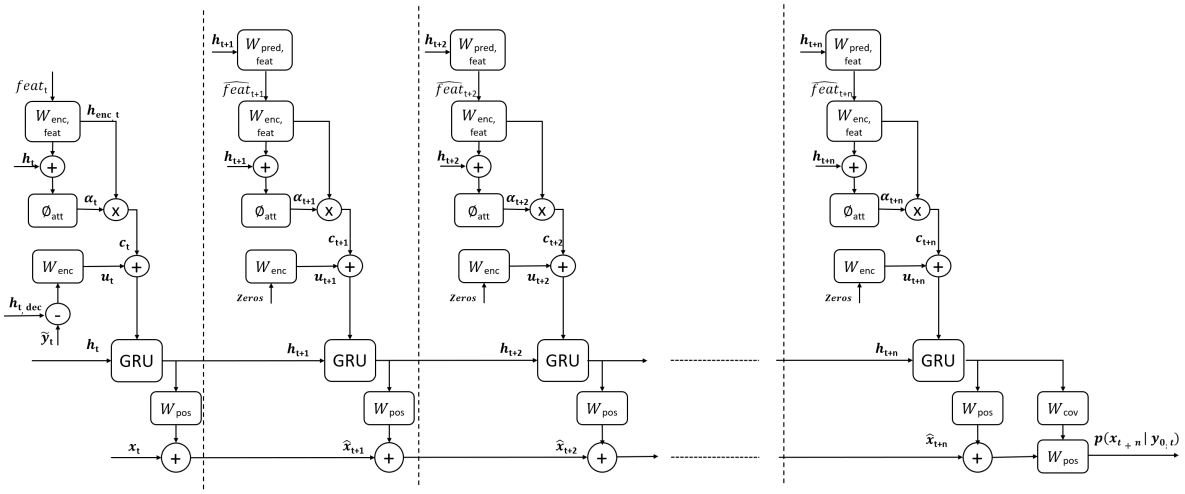

Figure 3.9: Prediction part: The future flow features are predicted by passing the hidden state of the GRU at current step through $W_{pred,net}$ and concatenated with encoded zero position vector. The network outputs a Gaussian distribution given by $p(x_{t+n}|y_{0:t})$ when predicting n steps ahead.

ture extractor, a fully connected layer would not be used in the final layer. The details of the CNN layers are as follows:

Operation	Filters	kernel	stride	Output
Convolution ReLU	8	3 × 3	1 × 1	8 × 64 × 32
Maxpooling	-	2 × 2	2 × 2	8 × 32 × 16
Convolution ReLU	8	3 × 3	2 × 2	8 × 16 × 8
Maxpooling	-	2 × 2	2 × 2	8 × 8 × 4

Table 3.1: Convolutional layers used as the feature encoder

In this network the soft visual attention proposed in [65] for caption generation from images is incorporated. The network would be taught to look at specific parts in the denseflow features more strongly. Compared to 'RNN_attn', the feature extractor is a CNN layer as shown in fig. 3.10. The 2nd layer of the CNN feature extractor produces L vectors of D-dimension, where $L = H \times W(8 \times 4)$ is the height and width of the output feature map and D the number of channels.

$$A = \{a_1, a_2, ..., a_L\},\ a_i \in \mathbb{R}^D \tag{3.23}$$

For visual attention the function ϕ_{attn} from eq. (3.20) is modified (ϕ_v) to take as input the extracted feature for different pixel location of the denseflow features in the form of annotation vectors a_i for $i = 1, 2, ...L$ which is the output of CNN layers and the hidden state to generate a weight which is indicative of the amount of focus the network should provide to that region to make future predictions.

$$\phi_v(h_t, a_i) = tanh\Big(\Big(W_h\, h_{t-1} + b_h\Big) + \Big(W_a\, a_i + b_a\Big)\Big) \tag{3.24}$$

$$\alpha_{t,i} = softmax\Big(\phi_v(h_t, a_i)\Big) \tag{3.25}$$

$$c_t = h_{enc,t} \,.\, \alpha_t \tag{3.26}$$

Where W_h, W_a, b_h and b_a are the weight and bias terms of a linear neural layer and c_t the context vector after element-wise multiplication of weights (α_t) with the encoded features ($h_{enc,t}$).

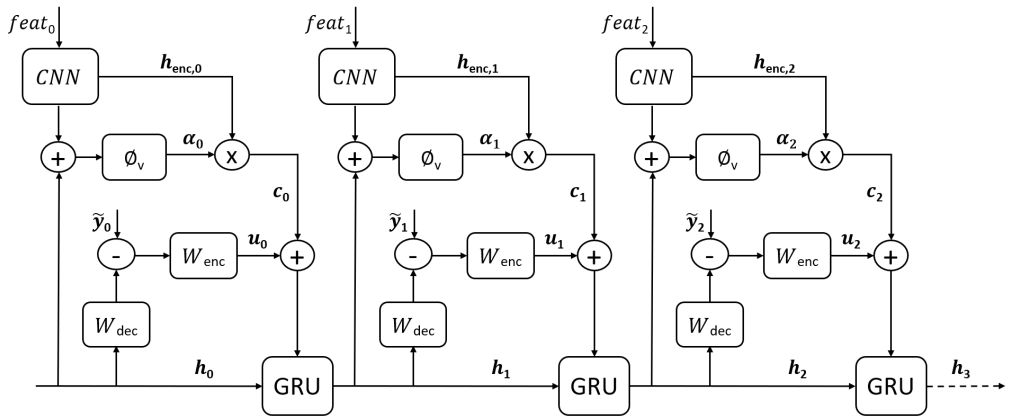

Figure 3.10: Convolution block is used to exploit the spatial information present in the denseflow features. The output from the convolutional block is weighted by soft-attention and concatenated with the output of the position encoder before feeding it to GRU for prediction.

3.4. Visualizing the attention areas

The visualisation of the focus areas learned by the models adds qualitative interpretability to understand where the model is looking at to make predictions. The input to the convolutional attention RNN is denseflow features with size 64×32 across both orientation and magnitude channels. In the final output of the CNN layers, the output has a dimension of 8×4 ($H \times W \times D$). The soft-attention weights are calculated on this output dimension, the receptive field of the final output layer corresponds spatially with the input dimension although they will be overlapping with each other. Thus to visualize the focus areas, the soft-attention weights will be upsampled by a factor of $2^3 = 8$ with Gaussian filter.

3.5. Coordinate system

Trajectory prediction methods have used two coordinate systems for final prediction: trajectory prediction made on the cameras image plane and predictions in the world coordinates. Trajectory prediction methods have used different benchmarks based on the final prediction coordinate system. Trajectory prediction in the image plane is made because 2D world coordinates are difficult to obtain. For world coordinate information, the dataset requires detection of the object in the LiDAR point cloud or through depth maps from stereo setups. Also, the existing benchmark dataset of curated tracks of pedestrian from a vehicles point-of-view such as JAAD [36] and PIE[48] are recorded from a monocular camera and hence lack the world coordinate information.

But there are particular challenges associated with making predictions in the image plane, first being that it is affected by the ego-vehicle motion. Position of a VRU in the image plane does not provide the AV with the exact information that can be used to plan for critical scenarios. Image plane information can be used to identify dangerous situations based on heuristics, but a lack of world coordinates information limits the possibility of using trajectory predictions to the planning part of the pipeline.

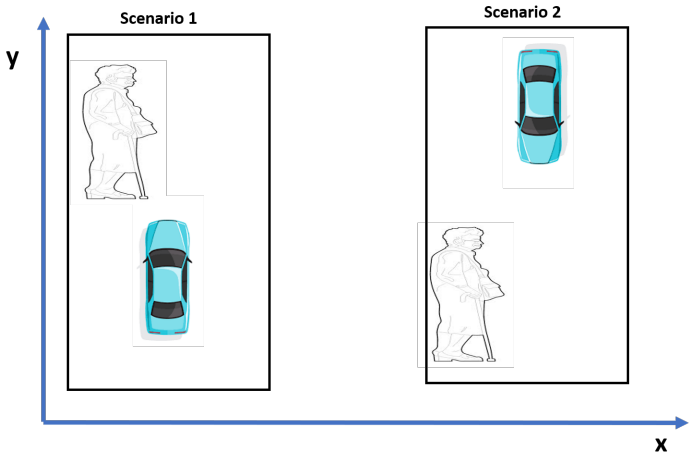

Figure 3.11: A sample scenario where the contextual cue in the 2 scenarios would be different, however the the pedestrian trajectory in world coordinates remains same

3.5.1. Predictions in the world coordinate system

Prediction in the 3D world would require accurate estimation in the detection and tracking blocks of the pipeline and aggregation of information from multiple sensors. LiDAR can provide the 3D coordinates of a VRU in ego-vehicle coordinates that can be converted to ego-motion compensated world coordinates provided there are accurate GPS/IMU measurements. The other practical aspect is the precise data association of the agents and tracks. However, this approach disentangles the future ego-motion of the vehicle from the trajectory prediction.

However, as shown in fig. 3.11, the contextual cue is evaluated from the ego-vehicle coordinate system, whereas the predictions are made in the world coordinate system. For example, if optical flow or hand gestures are evaluated in scenario 1 and 2, respectively in fig. 3.11, the contextual cue is different in both the scenarios; however the VRU trajectory would be same in world coordinate. The difference in the coordinate system of position and contextual cue will be rectified by transforming into the ego-vehicle coordinate frame where VRU was observed for the first time. Each track will have its coordinate system to which the subsequent observations will be transformed for predictions purposes and would be denoted by 'track coordinate'.

The transformation of a coordinate of a point given in a world coordinate system denoted by 'W' to the track coordinate system denoted by 'T' if it differs by an angle (fig. 3.12) is given by:

$$^{T}P = {}^{T}R_{W}\,{}^{W}P \tag{3.27}$$

If the track coordinate frame $o_1 x_1 y_1$ is oriented at an angle θ measured in anticlockwise direction to the world coordinate frame $o_0 x_0 y_0$, then the rotation matrix is given by:

$$^{T}R_{W} = \begin{pmatrix} cos(\theta) & sin(\theta) \\ -sin(\theta) & cos(\theta) \end{pmatrix} \tag{3.28}$$

Since 'track coordinate' system is the transformation to the ego-vehicle position at the first observation of VRU, this involves translation from world frame's origin to the track frames origin, followed by a rotation. Homogeneous coordinates allow embedding of both translation and rotation into a single 3×3 matrix given by:

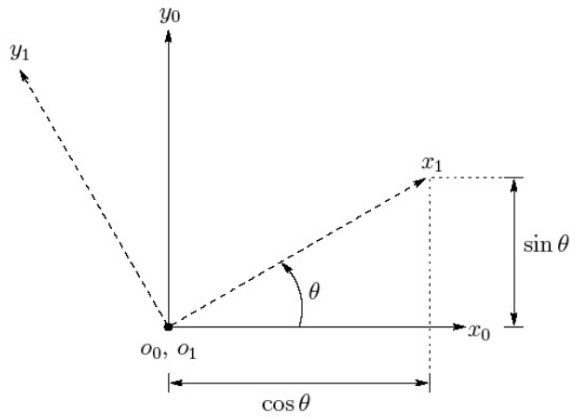

Figure 3.12: Track coordinate frame $o_1 x_1 y_1$ oriented at an angle θ to world coordinate frame $o_0 x_0 y_0$

$$ {}^{T}T_W = \begin{pmatrix} cos(\theta) & sin(\theta) & -P_x \\ -sin(\theta) & cos(\theta) & -P_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.29} $$

Where the origin of the 'track coordinate' system is at (P_x, P_y) with respect to the world coordinate frame.

3.6. Evaluation Metric

For the trajectory prediction in world-coordinates,geometric metrics are used to quantify the similarity between predicted trajectories and the ground truth and probablistic metrics to compare the accuracy and confidence in the future predictions.

3.6.1. Geometrical Accuracy

Accuracy of predicted trajectories can be measured by how closely they match with the actual path. Geometric accuracy basically measures this similarity. ADE (also called as Mean Squared Error) measures the dissimilarity by taking the mean of the euclidean distance between the predicted points and the actual path for a given prediction horizon.

3.6.2. Probabilistic Accuracy

Trajectory prediction algorithms are also able to express the confidence in future predictions by giving the output as a probabilistic distribution over future time instances. Metrics that compares the probabilistic distribution of predictions and the actual motion falls under this category. Probabilistic measures are indicative of both accuracy and confidence in the prediction.

Log-Likelihood (LL) is a a probabilistic accuracy metric that measures the expected likelihood of the ground truth observations given a predicted distribution. A method with a higher log likelihood will have a better fit of the predicted distribution with the ground truth. The Log-Likelihood of a track i with actual ground truth for predicting n steps ahead given the observation upto T is given by [43]:

$$ LL(i) = ln(P(x_{t+n}|x_{1:t})) \tag{3.30} $$

$$= -\frac{1}{2}\left[ln\Big(|\Sigma_{t+n}|\Big) + \Big(x_{t+n} - \hat{x}_{t+n}\Big)^T \Sigma_{t+n}^{-1}\Big(x_{t+n} - \hat{x}_{t+n}\Big) + k ln(2\pi) \right] \tag{3.31}$$

Where x_{t+n} is the ground truth position at n step ahead, $\hat{x}_{t+n}$ is the predicted position (mean of the Gaussian distribution) and the covariance matrix at $t+n$ step is given by Σ_{t+n} of order $k \times k$.

4

Experiments and results

This chapter describes the experiments to evaluate the performance of the proposed models with the addition of optical flow as a contextual feature. The experiments are performed on ECP-Dense dataset, the details of the processing of dataset is presented in section 4.1. The training and testing procedure for evaluating the performance of the model is elaborated in section 4.2. The experimental results and comparison of the performance of models are made in section 4.3. Finally, in section 4.4, the visualization of the focus regions in the denseflow using soft-attention is presented.

4.1. EuroCity Persons-Dense Dataset

EuroCity Persons (ECP) [9] is a large-scale naturalistic driving dataset recorded from a moving vehicle in multiple weather conditions in both day and night conditions. ECP dataset was recorded in 31 cities of 12 European countries, providing a broad data diversity in terms of location.

ECP-Dense dataset for path prediction and tracking is a subset of the much larger ECP dataset. The video sequences in the dataset is a curated selection of situations involving interesting or critical motions of a VRU in naturalistic traffic scenario. Images are annotated at 5 fps and in total contains 5233 tracks with atleast 2s of observations. Few of the sample scenes from the dataset are shown in fig. 4.1.

However, the raw dataset cannot be directly utilized for training and evaluating the models. The tracks are not labelled to provide a qualitative measure of their relative importance for the prediction task. Tracks of all the VRU visible during a video sequence are provided. However, one of the issues with raw ECP-Dense dataset is that tracks belonging to both pedestrian and riders are included in the dataset, and there is no label provided to separate them. In some cases, the ground truth position in world coordinates is inaccurate or anomalous, and these tracks need to be removed.

4.1.1. Processing of raw dataset

This section elaborates on the steps used to extract and process the ECP-Dense dataset before using it for experimentation. Following steps are undertaken in the pre-processing pipeline:

- From the raw ECP-Dense dataset, tracks with atleast 2s (10 frames) of continuous observations are curated. So, if there are missing observations in a specific track, they are divided such that parts of the track with atleast 2s of continuous observations is kept.

Figure 4.1: Sample scenes from ECP-Dense dataset

- Tracks that have a step size greater than 2m for 0.2s and those outside of 40m range from the ego-vehicle is removed. The range of 40 m is chosen as this is the reliable range for observation obtained from 3-D estimation of position from the tracking part of the pipeline for the ECP-Dense dataset.
- Riders are manually identified and labelled in the dataset.

After the pre-processing steps and removal of riders from the dataset, a total of 1697 tracks are available for training and evaluation, which would be denoted by ECP-D-tracks. The ECP-D-tracks is manually labelled into two categories based on the position and interaction of the pedestrian with the vehicle. The labelling is done to enable evaluation of performance in scenarios that are safety-critical for safe operation of an AV and relevant to the short term prediction horizon of 1s. The test set will only contain a subset of tracks belonging to 'Level 2' label whereas training set will contain tracks belonging to both 'Level 2' and 'Level 1' categories. The plot of the tracks belonging to 'Level 2' in the ECP-D-tracks is shown in fig. 4.3. The classification of tracks is based on the following rules:

- Level 2 (Nearby with interaction or a possibility of it): It includes tracks of a pedestrian crossing, waiting to cross, approaching the vehicle or walking along a narrow alley on which the autonomous vehicle needs accurate predictions for safe interaction. There are 876 tracks labelled under this category. The Level 2 tracks are further labelled by their motion type, as shown in fig. 4.2. Also, tracks with a visible jump in measurement are identified and labelled so as to remove it from the dataset.
- Level 1 (Far-away or not interacting): Tracks of pedestrian in ECP-D-tracks not falling under Level 2 are labelled under this category.

The tracks belonging to 'Level 2' are further classified into 3 types based on the motion types: standing; moving; or those with some dynamic changes in motion. While identifying the motion types, if a track is identified to contain measurement errors then those tracks are removed from the dataset. Also, the specific frame

where the motion changes occur in the 'Dynamic change' labelled tracks would be identified and labelled and would be indicated by Time To Event (TTE) = 0.

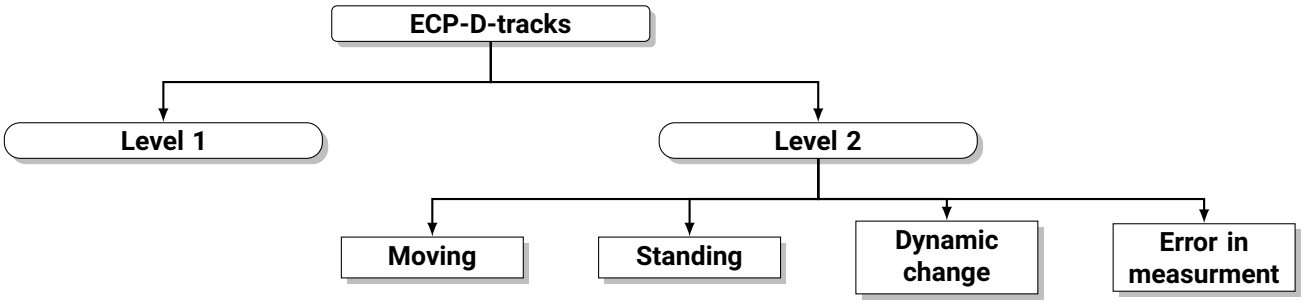

Figure 4.2: Labelling of ECP-D-tracks

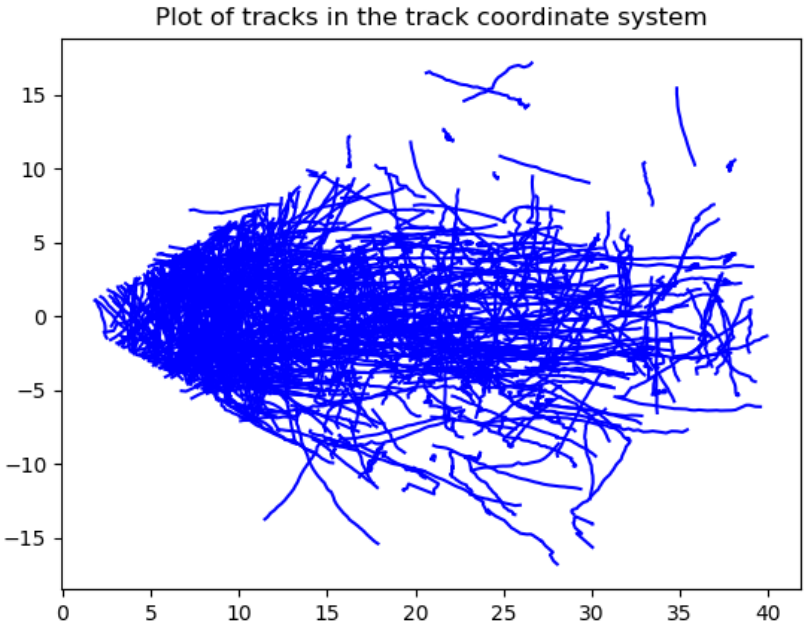

Figure 4.3: Figure shows the plot of tracks belonging to 'Level 2' category in track coordinate system

4.2. K-Fold Cross-validation

The accuracy of a machine learning model is evaluated by splitting the dataset into training and testing sets. The training set is used to train the model and testing set to evaluate the performance of the model. However, the accuracy obtained from a single test set can be biased. K-Fold cross-validation procedure is splitting the available data into multiple folds and ensuring that each fold is utilized as testing set at some point. During each evaluation, a random fold is selected for validation purposes and another for testing, and remaining k-2 folds are used for training (fig. 4.4). During each evaluation, the best performing model parameters based on performance on the validation set is saved and used to evaluate the performance on the test set. K-fold cross-validation results in a less optimistic estimate of the model accuracy compared to a simple train/test split.

For evaluating model accuracy, only 'Level 2' labelled tracks are used in the test set of k-fold as they involve

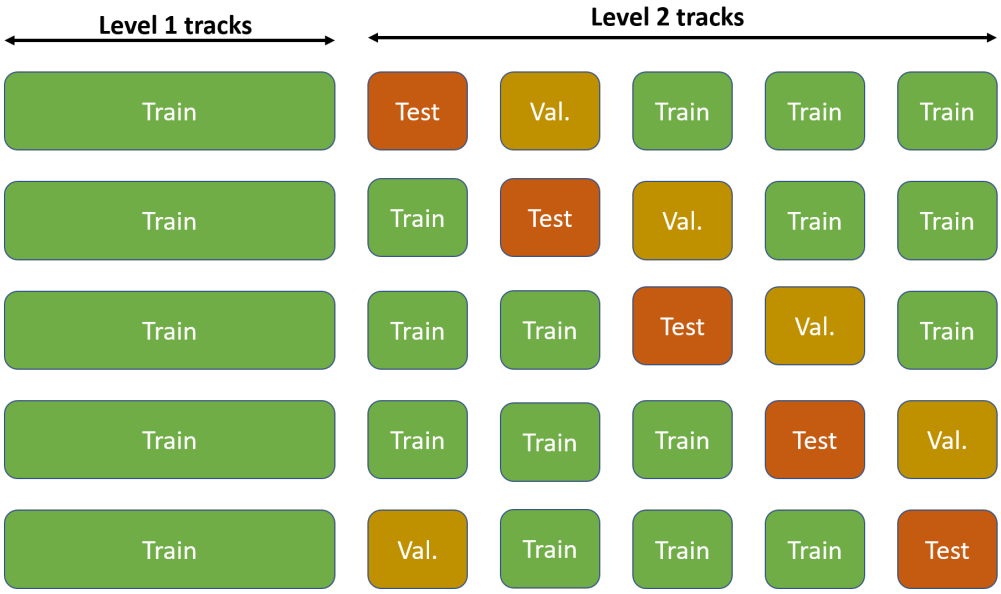

Figure 4.4: Splitting of the dataset into K-folds and holding out one split as validation and one as test set and the remaining k-2 splits along with 'Level 1' labelled tracks for training.

scenarios where the autonomous vehicle has to predict accurately for safe interaction. The procedure is as follows:

- The 'Level 2' labelled tracks are shuffled randomly and divided into k-folds.
- For each evaluation run, one of the folds is held out as validation set and another as test test. The remaining k-2 folds along with 'Level 1' labelled tracks will be used as the training set.

Table 4.1 gives the division of tracks into K-folds and the distribution of the type of tracks in each fold that would be used for evaluation purposes.

Fold	Moving	Standing	Dynamic change
Fold 1	115	22	23
Fold 2	104	23	33
Fold 3	100	22	30
Fold 4	102	26	31
Fold 5	106	23	30

Table 4.1: Number of tracks based on the motion type in different folds used as the test data

4.3. Results

This section evaluates the performance of the models and optical flow features proposed in chapter 3. The RNN models are trained using the Amsgrad [49] for 200 epochs, with a learning rate of 0.0003. The RNN is trained to minimize the log-likelihood loss of the predicted Gaussian distribution for the entire prediction horizon from 1 to n steps ahead. During training the hidden state output of the recurrent unit is reset to the initial state with a probability of 0.05 for every time step.

In section 4.3.1, the empirical comparison of the LSTM and GRU is given, based on which the choice of recurrent unit was done. In section 4.3.2 and section 4.3.3 the results in terms of log-likelihood and euclidean error is presented to compare the performances of proposed method.

4.3.1. Empirical comparison of recurrent units

The LSTM and GRU unit is compared for path prediction task by learning the probabilistic distribution over future predictions. For empirical comparison, 'Base_RNN' proposed in section 3.3.1 is utilized and the GRU recurrent block in fig. 3.6 and fig. 3.7 is replaced with LSTM when evaluating performance for that unit. Both GRU and LSTM is initialized with 64 elements in the hidden layer. The performance was evaluated using the k-fold cross validation process mentioned in section 4.2. Both GRU and LSTM based RNN network were trained on nvidia TITAN V GPU and it was found that GRU units reduced the training time by 6.81%. Also, as seen in table 4.2, the network with GRU as recurrent unit ('Base_RNN_gru') performs better than an LSTM based network for the entire prediction horizon.

Comparing both the training time and performance, the GRU unit performs better than an LSTM, the choice of recurrent unit is straightforward for this work. Therefore the models proposed in section 3.3.1, section 3.3.2 and section 3.3.3 utilizes GRU as the recurrent unit for modelling path prediction.

Model	Features		0.2s	0.4s	0.6s	0.8s	1.0s
Base_RNN_lstm	pos	mean	1.96	1.31	0.76	0.29	-0.2
		std	0.09	0.1	0.05	0.05	0.05
Base_RNN_gru	pos	mean	2.06	1.42	0.83	0.33	-0.17
		std	0.06	0.09	0.05	0.06	0.07

Table 4.2: Comparision of performance on using LSTM vs GRU as the fundamental block. The log-likelihood of predictions made on tracks belonging to 'Level 2' category shows that GRU has better performance in the entire prediction horizon

4.3.2. Log-likelihood results

The 'Level-2' tracks in ECP-D-tracks were further classified into 3 categories: 'moving', 'standing', and 'dynamic change' to evaluate the performance on various types of motion modes. In this section, the log-likelihood, which measures the goodness of the predicted Gaussian distribution for future time step with the actual ground truth, is evaluated for the combination of the proposed feature types and methods from section 3.1 and section 3.3 are presented in table 4.3 for the above-mentioned classification of motion types.

Baseline comparison using 'Base_RNN' model in table 4.3 across the 3 motion type shows that the 'Dynamic change' motion type tracks performed least favorably. For the 'Base_RNN' model with position (pos) as the only input, the log-likelihood for the tracks labelled as 'moving' or 'standing', the is 0.02 and 0.32 respectively for predicting 5 steps ahead (1s). However, for tracks that have some dynamic changes in the motion, the log-likelihood falls to -0.68 for predicting 5 steps ahead. Incorporating optical flow shows the steepest difference in performance for 1s ahead predictions compared 'Base_RNN' model with 'pos' as input for tracks with dynamic changes, with the 'Conv_attn' model performing the best with a log-likelihood of -0.37 (for 1s ahead). Comparison of performance of models on the tracks labelled as 'standing' shows that by incorporating 'Hist body' optical flow features, the performance is worse than a 'Base_RNN' with only 'pos' as input for predicting 1s ahead. However, for predicting 0.2s and 0.4s ahead 'Hist body' performs slightly better. Even with 'pos + Hist cell' and 'pos + Denseflow' features, the difference in log-likelihood performance (with only 'pos' as input) decreases as the prediction horizon increases to 1s. One of the reasons for this could be attributed to the fact that out of the tracks under 'Level 2', the tracks in which the pedestrian is standing accounts for only 14.7% leading to a comparatively smaller number of training data available under this category. Comparison across all 3 categories shows that, as a general trend the performance increases from 'Base_RNN' to 'RNN_attn' with 'pos + Hist body' (except for tracks of a pedestrian standing still), 'RNN_attn' with 'pos + Hist cell' and finally to 'Conv_attn' with 'pos + Denseflow' features.

Track Label	Model	Features		0.2s	0.4s	0.6s	0.8s	1.0s
Moving	Base_RNN	pos	mean	2.16	1.53	0.96	0.46	0.02
			std	0.1	0.1	0.11	0.11	0.11
	RNN_attn	pos+ Hist body	mean	2.29	1.58	0.97	0.45	-0.01
			std	0.13	0.08	0.09	0.08	0.07
	RNN_attn	pos+ Hist cell	mean	2.39	1.68	1.06	**0.54**	0.08
			std	0.14	0.107	0.12	0.1	0.1
	Conv_attn	pos + Denseflow	mean	**2.42**	**1.7**	**1.07**	**0.54**	**0.1**
			std	0.13	0.1	0.12	0.11	0.12
Standing	Base_RNN	pos	mean	2.21	1.58	1.1	0.67	0.32
			std	0.2	0.28	0.26	0.25	0.26
	RNN_attn	pos+ Hist body	mean	2.39	1.63	1.1	0.65	0.24
			std	0.29	0.3	0.26	0.23	0.24
	RNN_attn	pos+ Hist cell	mean	2.5	1.78	1.24	0.77	0.37
			std	0.18	0.19	0.18	0.18	0.18
	Conv_attn	pos+ Denseflow	mean	**2.51**	**1.82**	**1.3**	**0.85**	**0.44**
			std	0.2	0.2	0.16	0.15	0.16
Dynamic change	Base_RNN	pos	mean	2.08	1.28	0.6	-0.03	-0.68
			std	0.23	0.26	0.25	0.22	0.28
	RNN_attn	pos+ Hist body	mean	2.25	1.32	0.6	-0.03	-0.57
			std	0.22	0.22	0.22	0.22	0.25
	RNN_attn	pos+ Hist cell	mean	**2.35**	1.44	0.7	0.07	-0.46
			std	0.2	0.18	0.18	0.16	0.17
	Conv_attn	pos + Denseflow	mean	2.34	**1.45**	**0.74**	**0.13**	**-0.37**
			std	0.24	0.2	0.25	0.23	0.2

Table 4.3: The log-likelihood of predictions made on tracks belonging to 'Level 2' category and different type of motions. The highlighted results shows the best performing model for the corresponding motion type and prediction horizon. A major difference in performance from the baseline model when predicting exactly 1s ahead is observed for the 'Dynamic change' labelled tracks. Otherwise the difference in log-likelihood of predictions between the baseline and optical flow feature model starts to decrease as the prediction horizon increases.

In fig. 4.5, the log-likelihood of predictions for tracks with dynamic changes is plotted for TTE = -10 to TTE = 5. In the figure, the log-likelihood at a particular frame is evaluated for making predictions 1s ahead (5steps). TTE = 0 indicates the labelled frame at which some sort of dynamic change in the motion is observed. From fig. 4.5, it can be seen that the performance of all the models with their respective input features have similar performance for TTE = -10 to TTE = -6 and TTE = 2 to TTE = 5. Between TTE = -5 to TTE = 1, 'pos+Denseflow' features, which are fed to the 'Conv_attn' model has the best performance, followed by 'pos+Hist cell' and 'pos+Hist body'. Incorporating optical features have shown that the performance in terms of log-likelihood improves predictions made during the dynamic changes from one motion mode to the other compared to a 'Base_RNN' that utilizes only positional features.

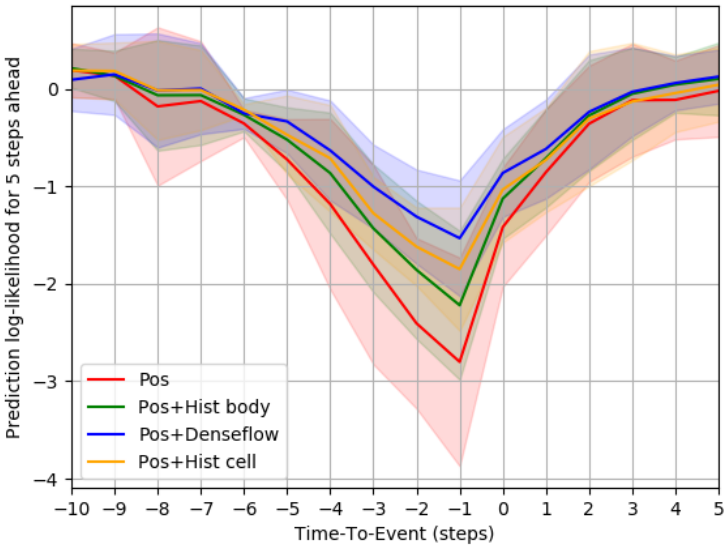

Figure 4.5: For predicting exactly 1s ahead (5 steps), the mean (solid lines) and standard deviation (shaded area) of the log-likelihood of the predictions for TTE = -10 to TTE = 5 for tracks with dynamic changes. 'pos+Denseflow' features have a comparatively better log-likelihood on predictions made from TTE=-5 to TTE = 1.

4.3.3. Comparison of predictions

As can be seen table 4.4, comparing the euclidean error on the predicted positions, the difference between models is comparatively very small. In table 4.4, the average euclidean error and the lateral position error on the entire track looks similar across all the considered input features. Even with the addition of optical flow features, the reduction in error is ≈ 1cm at best. To look further into this, in table 4.5, comparison is made over scenarios with changes in the motion type. The euclidean and lateral error of predictions for 1s (5 steps) ahead made at TTE = -2 to TTE = +2 also shows that only a slight improvement of 2cm is seen for denseflow features features from TTE = -2 to TTE = 0. Following this, the performance of 'Pos + Hist cell' and 'pos + Denseflow' features slightly decreases at TTE = +2 in comparison to the baseline model.

Figure 4.6 compares the performance of the optical flow feature (Hist body, Hist cell and denseflow) and the baseline model for a sample track where the pedestrian turns and walks in front of the vehicle. The figure shows that there is a difference in the predicted confidence even though the predicted path is similar for the models. Figure 4.6 shows the uncertainty in terms of 2-sigma interval ellipse around the 1s ahead (5 steps) predicted position starting from TTE = -5 to TTE = +1. Predictions made at TTE = -5 corresponds to the future position at which the pedestrian is observed to start turning. At TTE = -5, 'pos + Hist body' model has the highest uncertainty with 'pos + Denseflow' having a well-rounded and equal uncertainty along both directions. It may be noted that between TTE = -5 to TTE = 0, the positional input feature shows no changes and hence for TTE = -5, -3 and -1 in fig. 4.6, the 'pos' model in fact gets more confident and as no major changes in the position is observed to indicate a turning behaviour. However, models with 'pos+ Hist body', 'pos+ Hist cell', and 'pos + Denseflow' clearly shows an elongation of uncertainty along the lateral direction. This indicates that the incorporation of optical flow features is helping the model detect the changes in the motion type.

It may also be noted that under 'Dynamic change' labelled tracks, there are multiple types of behaviour: stopping, starting to move, turning, etc. and RNN model has comparatively fewer training samples to learn multiple behaviours. For example, the tracks labelled under 'Dynamic change' accounts for only 18.6% of

tracks in ‘Level 2’ and even within that only a few frames are indicative of the changes in motion behaviour.

Model	Features		0.2s	0.4s	0.6s	0.8s	1.0s
Base_RNN	pos	L2 (m)	0.09	0.14	0.2	0.27	0.34
		lat.(m)	0.05	0.09	0.13	0.17	0.22
RNN_attn	pos+ Hist body	L2 (m)	0.08	0.14	0.2	0.26	0.33
		lat.(m)	0.05	0.08	0.13	0.17	0.22
RNN_attn	pos+ Hist cell	L2 (m)	0.08	0.14	0.2	0.26	0.33
		lat.(m)	0.05	0.08	0.12	0.16	0.21
Conv_attn	pos+ Denseflow	L2 (m)	0.08	0.14	0.2	0.26	0.33
		lat.(m)	0.05	0.08	0.12	0.16	0.21

Table 4.4: The average euclidean (L2) and lateral error (lat.) of predictions made on tracks belonging to ‘Level 2’ category are given in meters. All the models have a similar type of performance in this metric

Track Label	Model	Features		TTE=-2 1.0s	TTE=-1 1.0s	TTE= 0 1.0s	TTE=+1 1.0s	TTE=+2 1.0s
Dynamic change	Base_RNN	pos	L2 (m) lat.(m)	0.47	0.49	0.46	0.4	0.36
	RNN_attn	pos+ Hist body	L2 (m) lat.(m)	0.47	0.49	0.46	0.4	0.36
	RNN_attn	pos+ Hist cell	L2 (m) lat.(m)	0.46	0.49	0.46	0.39	0.35
	Conv_attn	pos+ Denseflow	L2 (m) lat.(m)	0.45	0.47	0.45	0.41	0.37

Table 4.5: The average euclidean and lateral error for predictions made from TTE = -2 to TTE = +2 is presented and given in meters.

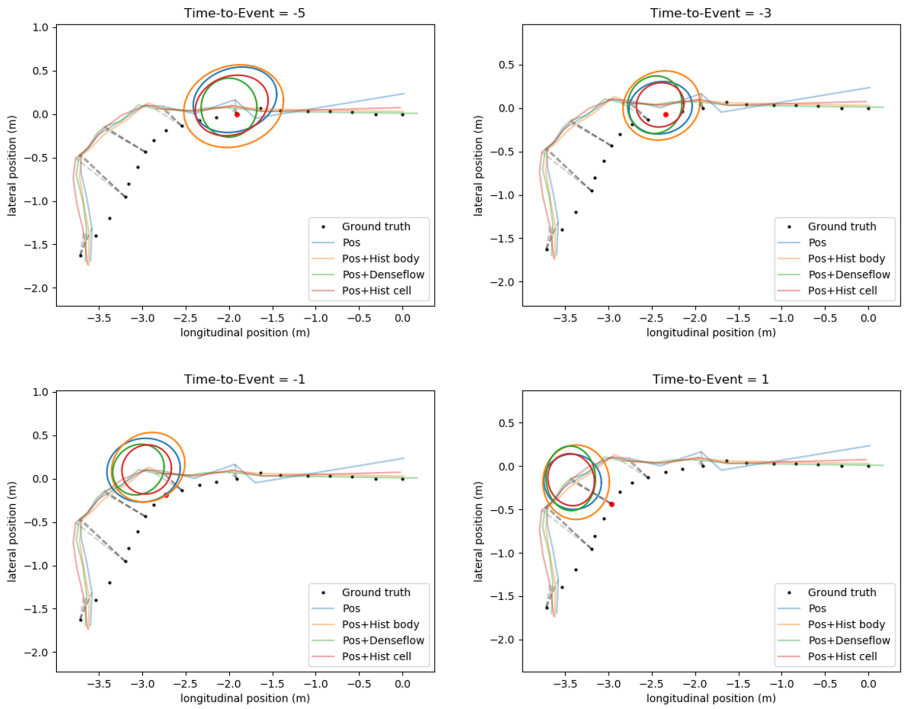

Figure 4.6: The plots show the prediction made exactly 1s ahead (5 steps) for TTE = -5, -3, -1 and 0. The 5 steps ahead prediction at TTE = -5 corresponds to the frame at which the dynamic change in motion mode occurs. The ellipse around the predicted point shows the 2-sigma uncertainity region. The point marked with red in each of the plot corresponds to the ground truth for the 1s ahead prediction corresponding to the TTE.

4.4. Qualitative evaluation

This section describes the visualisation of the attention weight that the 'Conv_attn' model assigns to dense optical flow for some of the scenarios. Figure 4.7 - fig. 4.9 show the attention weight map where the network is concentrating in the dense optical flow for sample tracks from 'moving', 'standing' and 'Dynamic change' labels. It can be observed that for the sample track shown in fig. 4.7 for a pedestrian crossing the road, the attention weight at the start of the track focuses on the upper body and head, and over the course of the track the focus shifts towards legs. In fig. 4.8, the pedestrian is standing at one place and hence lacks relative motion in legs; however, it can be seen that the attention weight is concentrated mainly on the upper torso and closer to the head. The attention weight is also affected by the movement of another pedestrian in the background (see Frame(Fr) = 5 in fig. 4.8) the weight is also assigned to the background motion. Figure 4.9 shows the visualization of dense optical flow and the attention weight map right before the time a pedestrian starts to make a turn, which clearly shows that the network is looking closely at the leg and the head.

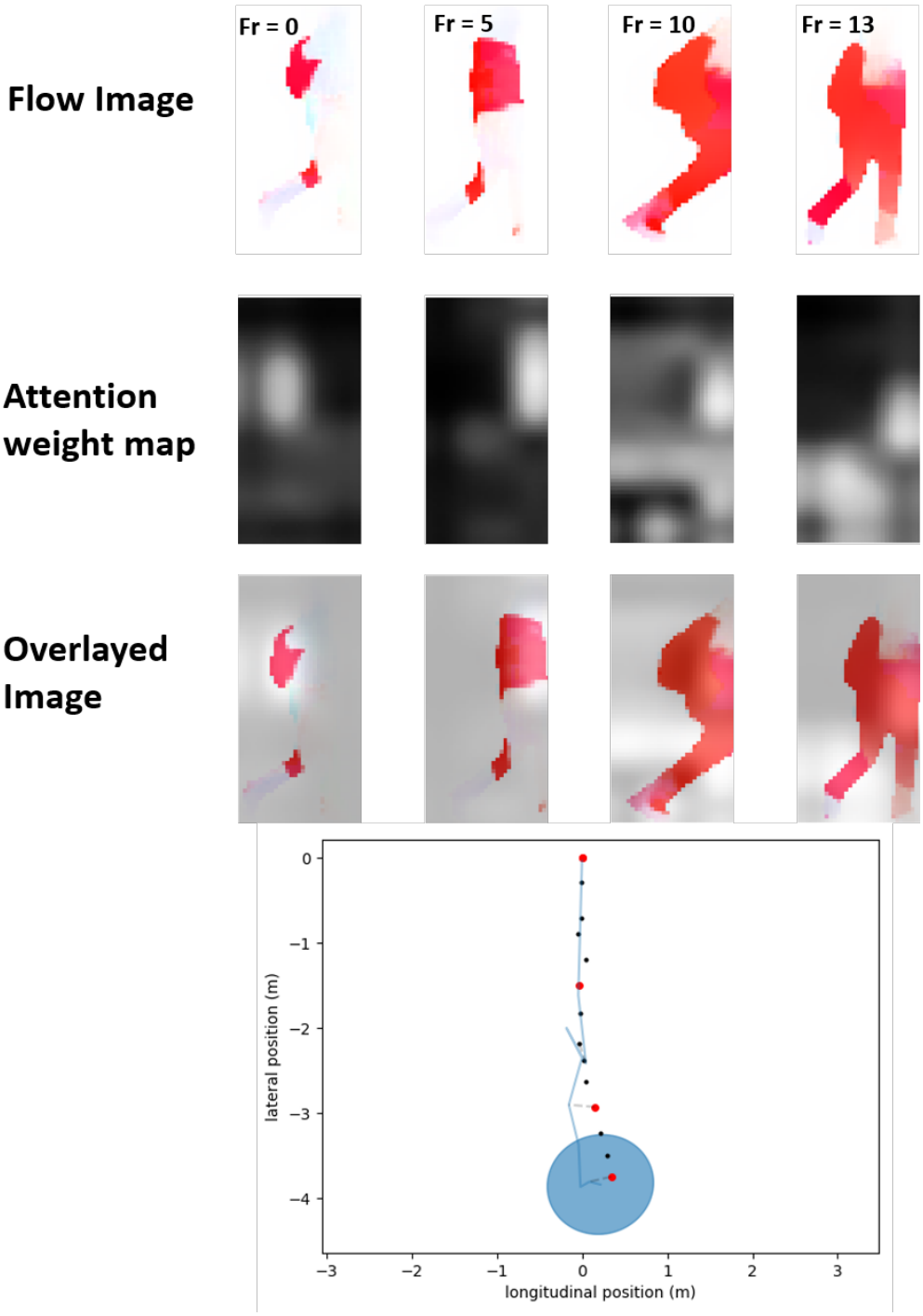

Figure 4.7: Figure shows the attention weights indicating where exactly the network is looking in the dense optical flow. In the attention weight map, the region with lighter shade have more weight. In the overlayed image, the attention weight map is plotted over the dense optical flow and regions with white blobs is indicative of the relevant areas the network is paying more attention to. Attention weight is given for frames(Fr) 0, 5, 10 and 13. In the plot of the track, points marked with red dot indicates the frames for which the attention weights is shown. Plot of the predictions by 'Conv_attn' model is given by the solid lines. Track start at (0,0) for Frame (Fr) = 0. The 2-sigma uncertainty ellipse is plotted for the final position. Attention weight shows that the focuses of the network shifts from upper body towards legs as the pedestrian moves forward.

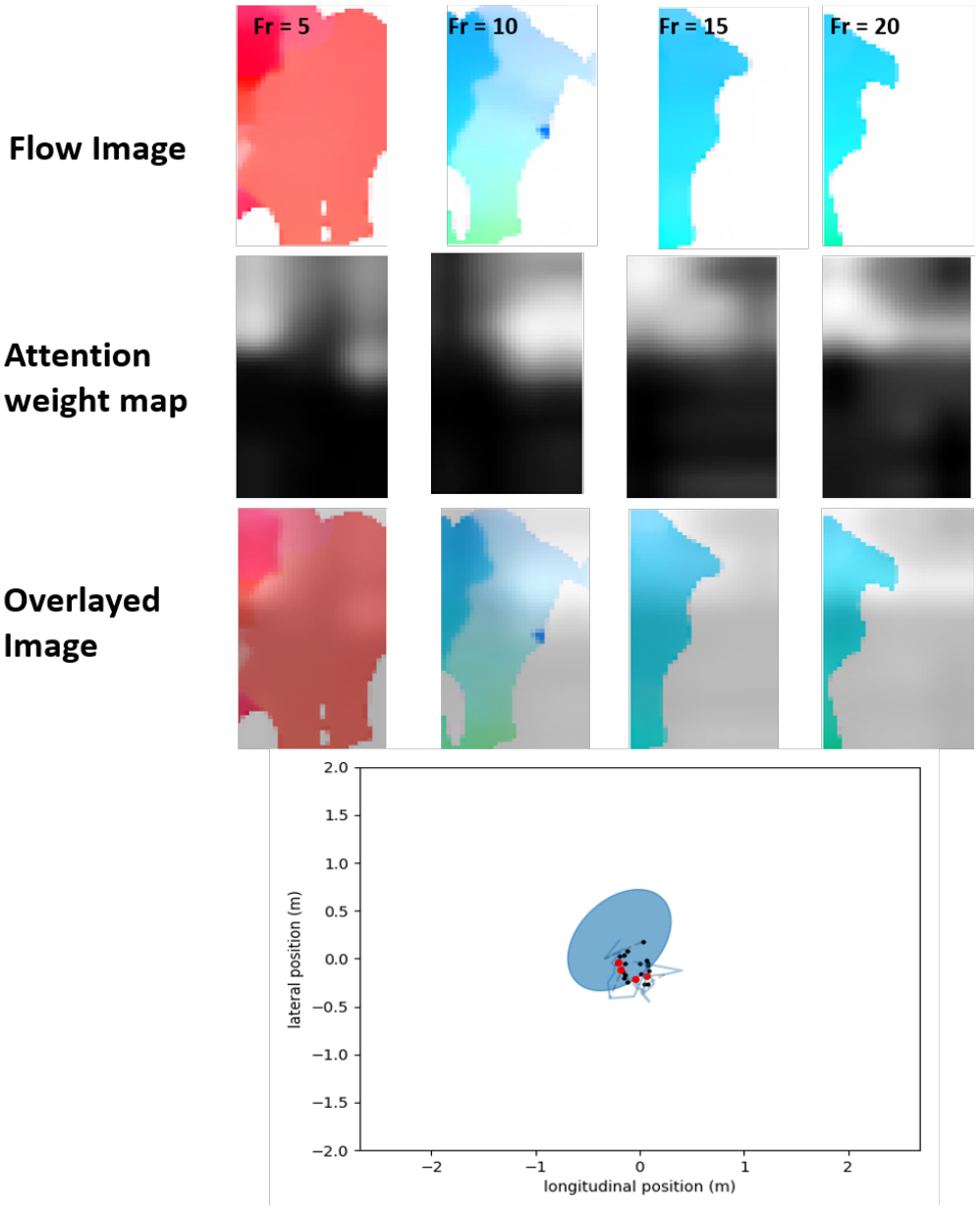

Figure 4.8: For a pedestrian standing at one place, the attention weight in this case is heavily focused on the region near the head; however, at Frame (Fr) = 0, it is affected by the background motion of another pedestrian. Attention weight is given for frames(Fr) 0, 5, 10, 15 and 20. In the attention weight map and the Overlayed image, the regions with white blobs indicates more focus or weights. Track start at (0,0) for Frame (Fr) = 0 and the frames at which attention is shown is indicated by red dot in the track plot at the bottom.

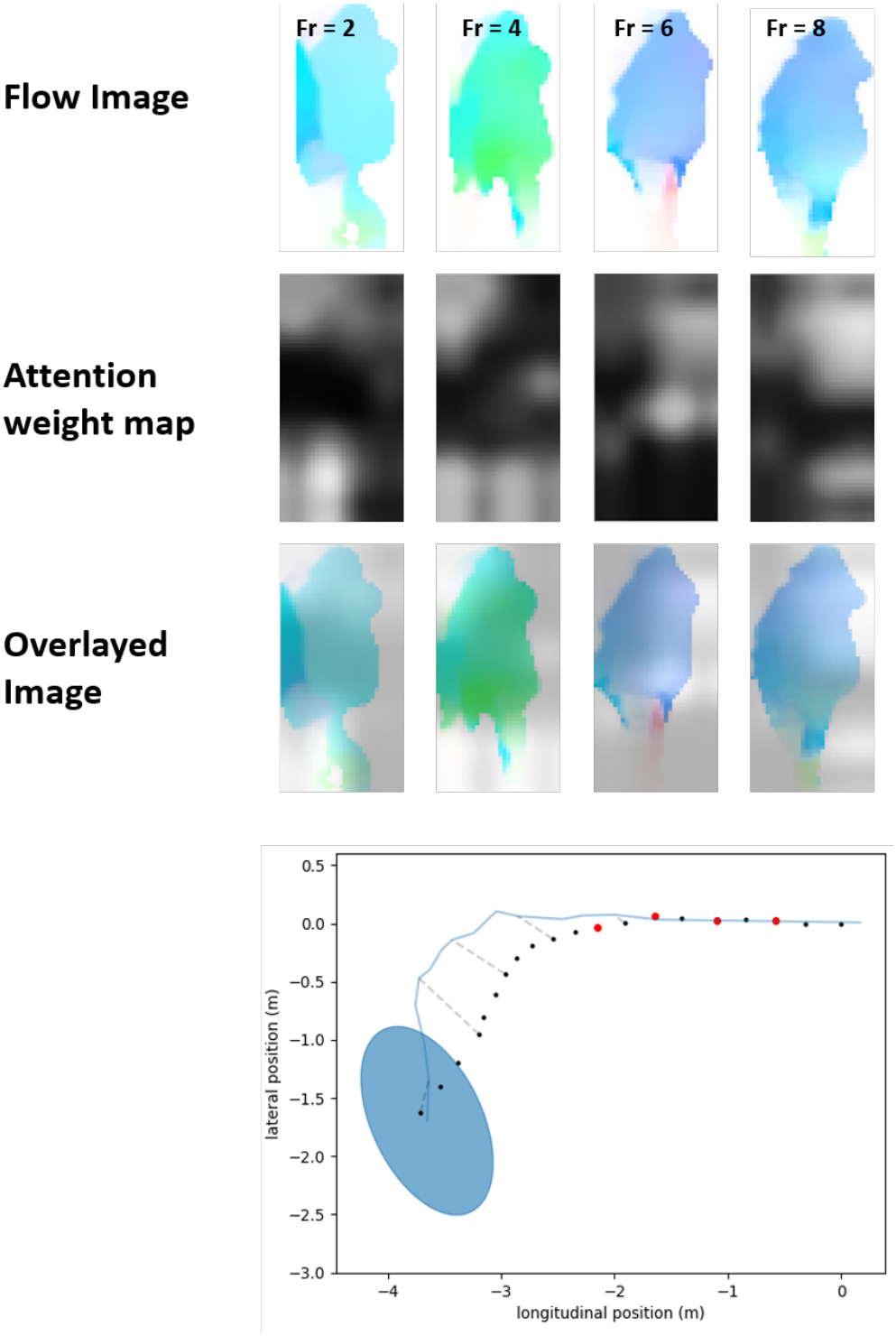

Figure 4.9: For one of the sample tracks with pedestrian turning and moving in front of the vehicle, attention weights is plotted for frames right before the turning behaviour is observed at Frame (Fr) = 6. The attention weight map shows that the network is looking closely at both leg and the head. Track start at (0,0) for Frame (Fr) = 0 and the frames at which attention is shown (Fr 2,4,6,8) is indicated by red dot in the track plot at the bottom.

5

Conclusions

This thesis evaluated the benefits of incorporating optical flow as a contextual cue to improve path prediction of pedestrians in a naturalistic driving dataset. The benefits of using high-dimensional dense optical flow in comparison to the lower-dimensional histogram features, the benefits of retaining high-dimensional features and utilizing the spatial relations through a convolutional-attention network have been evaluated. The convolutional-attention network, inspired by the visual attention method proposed in [65], was used for caption generation from images by concentrating on relevant areas in the image. The design idea was to use a convolutional-attention network to: (a) identify the relevant pixels and regions in the dense optical flow that would affect the future motion of pedestrians; and (b) provide a qualitative step to add interpretability to the RNN model utilizing dense optical flow features. Experiments were performed on the proposed combination of methods and extracted optical flow features, to answer the research question from section 1.4. The conclusions drawn from this work are presented in section 5.1 - section 5.3. The performance was evaluated on ECP-Dense, which is a naturalistic driving dataset containing curated scenarios that would be of interest for path prediction tasks. Further discussion on taking this work further is presented in section 5.4.

5.1. What is a suitable method and approach to model path prediction?

With the availability of large scale datasets, recent development and adoption of deep learning methods; the non-linear regression methods making use of RNN have been the state-of-art for trajectory prediction task in benchmarks like JAAD, PIE, and Stanford drone dataset. Comparison of performance of methods from the literature on path prediction task on the benchmarks was made in section 2.5. On Stanford drone dataset, the best performing method is Sophie [54], which combines RNN with attention and Generative adversarial network to propose socially and physically constrained pedestrian trajectories. In [44], a comparison between DBN and RNN has shown that the latter is able to outperform the DBN for predicting the cyclist track in cases where it is either turning or moving straight. The other advantage is that the RNN network allows for the flexibility to incorporate high-dimensional contextual cues unlike in a clustering-based approach, whose real-time performance would be affected as the input dimension increases. Also, unlike methods such as GPDM, RNN based methods can scale as the amount of training data increases.

Based on the conclusions drawn from benchmarks and previous work, RNN based method was chosen to model path prediction by incorporating optical flow features. In section 4.3.1, the choice of recurrent units

in terms of either an LSTM or GRU was empirically evaluated. It was found that the GRU recurrent unit performed better both in terms of prediction accuracy and training time.

5.2. How does the accuracy in terms of probabilistic confidence and predicted position affected by the incorporation of optical flow?

Based on the results from section 4.3.2, it can be seen that in general, the performance across various motion types increases with the incorporation of optical flow features. The 'pos+Denseflow' features incorporated into the 'Conv_attn' gives better results. The major improvement in performance was seen with tracks that had some dynamic changes in the motion types. Furthermore, the performance in terms of log-likelihood of the predictions improved with optical flow features, when a change in motion mode was observed for a track. However, predictions in terms of euclidean error showed only a slight improvement with 'pos+Denseflow' improving the prediction accuracy by $\approx$ 2 cm for TTE = -2 to TTE = 0.

5.3. Is dimensionality reduction needed to make dense optical flow a feasible contextual cue?

In this work, a comparison was made by reducing the dimension of optical flow features of pedestrians into histograms [32] and using the dense optical flow features. The model suggested in section 3.3.3 utilizes a convolutional network to extract contextual information from the dense optical flow features keeping the shape, structure, and spatial constraints, and soft-attention is applied to the extracted features to identify the relevant regions where the model should look closely into. Comparison of the histogram features (Hist body and Hist cell) and dense optical flow features with 'Conv_attn' model in section 4.3.2 has shown that the latter can be directly fed to the RNN network without loss in information or performance.

5.4. Future work

In this work path prediction of a pedestrian was done by processing a naturalistic driving dataset and incorporating optical flow as contextual cue. Building upon this, future work that can done in this directions are:

- A naturalistic driving dataset contains a lot of tracks, however, it suffers from a class imbalance issue for frames that are indicative of dynamic changes. For short term predictions on the dataset, modification of training methodology by adding synthetic data or penalizing the model more on dynamic changes can be tried out.
- From an application perspective, handling of missing frames in the tracks in the modelling prediction task is a challenge that needs to be addressed. In this work, continuous tracks were created by splitting the tracks at the point when an observation was missing.
- The visualisation of the attention weights of the RNN model on dense optical flow can be compared with the individual hand-crafted cues at frames where they are indicative of changes in the motion type.
- ECP-Dense dataset can be a good benchmark for the path prediction task, and few of the processing steps to extract relevant tracks from this work can be adopted, and additional labels on intention and type of behaviour expected at each frame identified by human annotators can be included with the tracks.

Bibliography

[1] Waymo open dataset: An autonomous driving dataset, 2019.

[2] Alexandre Alahi, Vignesh Ramanathan, and Li Fei-Fei. Socially-aware large-scale crowd forecasting. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2203–2210, 2014.

[3] 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*, pages 961–971, 2016.

[4] Ben Benfold and Ian Reid. Guiding visual surveillance by tracking human attention. In *Proceedings of the 20th British Machine Vision Conference*, volume 2, page 7, 2009.

[5] Apratim Bhattacharyya, Mario Fritz, and Bernt Schiele. Long-term on-board prediction of people in traffic scenes under uncertainty. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4194–4202, 2018.

[6] Jiang Bian, Dayong Tian, Yuanyan Tang, and Dacheng Tao. A survey on trajectory clustering analysis. *arXiv preprint arXiv:1802.06971*, 2018.

[7] Julian Bock, Robert Krajewski, Tobias Moers, Steffen Runde, Lennart Vater, and Lutz Eckstein. The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections. *arXiv preprint arXiv:1911.07602*, 2019.

[8] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*, 2012.

[9] Markus Braun, Sebastian Krebs, Fabian Flohr, and Dariu Gavrila. Eurocity persons: a novel benchmark for person detection in traffic scenes. *IEEE transactions on pattern analysis and machine intelligence*, 41 (8):1844–1861, 2019.

[10] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 11621–11631, 2020.

[11] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 8748–8757, 2019.

[12] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[13] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[14] William F Clocksin. Determining the orientation of surfaces from optical flow. In *Proceedings of the 1978 AISB/GI Conference on Artificial Intelligence*, pages 93–102. IOS Press, 1978.

[15] Navneet Dalal. *Finding people in images and videos*. PhD thesis, 2006.

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[17] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of IEEE International conference on Computer Vision*, pages 2758–2766, 2015.

[18] Andreas Ess, Bastian Leibe, Konrad Schindler, and Luc Van Gool. A mobile vision system for robust multi-person tracking. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[19] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

[20] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.

[21] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. In *International Conference on Artificial Neural Networks*, pages 850–855, 1999.

[22] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2255–2264, 2018.

[23] Joko Hariyono, Ajmal Shahbaz, and Kang-Hyun Jo. Estimation of walking direction for pedestrian path prediction from moving vehicle. In *IEEE/SICE International Symposium on System Integration (SII)*, pages 750–753, 2015.

[24] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5): 4282, 1995.

[25] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[26] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[27] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 8981–8989, June 2018.

[28] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2462–2470, 2017.

[29] SAE International. Automated driving levels of driving automation are defined in new sae international standard j3016.

[30] Ramesh Jain, WN Martin, and JK Aggarwal. Segmentation through the detection of changes due to motion. *Computer Graphics and Image Processing*, 11(1):13–34, 1979.

[31] Robert Kaucic, AG Amitha Perera, Glen Brooksby, John Kaufhold, and Anthony Hoogs. A unified framework for tracking through occlusions and across sensor gaps. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, volume 1, pages 990–997, 2005.

[32] Christoph G Keller and Dariu M Gavrila. Will the pedestrian cross? a study on pedestrian path prediction. *IEEE Transactions on Intelligent Transportation Systems*, 15(2):494–506, 2013.

[33] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet. Lyft level 5 av dataset 2019, 2019. URL `https://level5.lyft.com/dataset/`.

[34] Kris M Kitani, Brian D Ziebart, James Andrew Bagnell, and Martial Hebert. Activity forecasting. In *Proceedings of the European Conference on Computer Vision*, pages 201–214. Springer, 2012.

[35] Julian F. P. Kooij, Fabian Flohr, Ewoud A. I. Pool, and Dariu M. Gavrila. Context-based path prediction for targets with switching dynamics. *International Journal of Computer Vision*, 127(3):239–262, Mar 2019.

[36] Iuliia Kotseruba, Amir Rasouli, and John K Tsotsos. Joint attention in autonomous driving (jaad). *arXiv preprint arXiv:1609.04741*, 2016.

[37] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip HS Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 336–345, 2017.

[38] Xiaofei Li, Fabian Flohr, Yue Yang, Hui Xiong, Markus Braun, Shuyue Pan, Keqiang Li, and Dariu M Gavrila. A new benchmark for vision-based cyclist detection. In *IEEE Intelligent Vehicles Symposium*, pages 1028–1033, 2016.

[39] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[40] Wei-Chiu Ma, De-An Huang, Namhoon Lee, and Kris M Kitani. Forecasting interactive dynamics of pedestrians with fictitious play. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 774–782, 2017.

[41] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[42] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *Proceedings of IEEE International conference on Computer Vision*, pages 261–268, 2009.

[43] Ewoud AI Pool, Julian FP Kooij, and Dariu M Gavrila. Using road topology to improve cyclist path prediction. In *IEEE Intelligent Vehicles Symposium*, pages 289–296, 2017.

[44] Ewoud AI Pool, Julian FP Kooij, and Dariu M Gavrila. Context-based cyclist path prediction using recurrent neural networks. In *IEEE Intelligent Vehicles Symposium*, pages 824–830, 2019.

[45] R. Quintero, J. Almeida, D. F. Llorca, and M. A. Sotelo. Pedestrian path prediction using body language traits. In *IEEE Intelligent Vehicles Symposium*, pages 317–323, June 2014.

[46] A. Rasouli, I. Kotseruba, and J. K. Tsotsos. Are they going to cross? a benchmark dataset and baseline for pedestrian crosswalk behavior. In *IEE International Conference on Computer Vision Workshops (ICCVW)*, pages 206–213, Oct 2017.

[47] A. Rasouli, I. Kotseruba, and J. K. Tsotsos. Understanding pedestrian behavior in complex traffic scenes. *IEEE Intelligent Vehicles Symposium*, 3(1):61–70, March 2018.

[48] Amir Rasouli, Iuliia Kotseruba, Toni Kunic, and John K Tsotsos. Pie: A large-scale dataset and models for pedestrian intention estimation and trajectory prediction. In *Proceedings of IEEE International conference on Computer Vision*, pages 6262–6271, 2019.

[49] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *Proceedings of International Conference on Learning Representations*, 2018.

[50] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese. Learning social etiquette: Human trajectory prediction in crowded scenes. In *Proceedings of the European Conference on Computer Vision*, 2016.

[51] Christoph Rösmann, Malte Oeljeklaus, Frank Hoffmann, and Torsten Bertram. Online trajectory prediction and planning for social robot navigation. In *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 1255–1260, 2017.

[52] 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.

[53] Amir Sadeghian, Ferdinand Legros, Maxime Voisin, Ricky Vesel, Alexandre Alahi, and Silvio Savarese. Car-net: Clairvoyant attentive recurrent network. In *Proceedings of the European Conference on Computer Vision*, pages 151–167, 2018.

[54] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1349–1358, 2019.

[55] Hojjat Salehinejad, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaee. Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*, 2017.

[56] S Srikanth, JA Ansari, RK Ram, S Sharma, JK Murthy, and KM Krishna. Infer: Intermediate representations for future prediction. In *International Conference on Intelligent Robots and Systems*, 2019.

[57] Russell Stuart, Norvig Peter, et al. Artificial intelligence: a modern approach, 2003.

[58] Oily Styles, Arun Ross, and Victor Sanchez. Forecasting pedestrian trajectory with machine-annotated training data. In *IEEE Intelligent Vehicles Symposium*, pages 716–721, 2019.

[59] Deqing Sun, Stefan Roth, and Michael J Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *International Journal of Computer Vision*, 106(2):115–137, 2014.

[60] I Sutskever, O Vinyals, and QV Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[61] Jur Van den Berg, Ming Lin, and Dinesh Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *IEEE International Conference on Robotics and Automation*, pages 1928–1935, 2008.

[62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[63] Anirudh Vemula, Katharina Muelling, and Jean Oh. Social attention: Modeling attention in human crowds. In *IEEE International Conference on Robotics and Automation*, pages 1–7, 2018.

[64] Jack Wang, Aaron Hertzmann, and David J Fleet. Gaussian process dynamical models. In *Advances in neural information processing systems*, pages 1441–1448, 2006.

[65] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[66] Stefan Zernetsch, Viktor Kress, Bernhard Sick, and Konrad Doll. Early start intention detection of cyclists using motion history images and a deep residual network. In *IEEE Intelligent Vehicles Symposium*, pages 1–6, 2018.

[67] Shanshan Zhang, Rodrigo Benenson, and Bernt Schiele. Citypersons: A diverse dataset for pedestrian detection. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3213–3221, 2017.

[68] Tianyang Zhao, Yifei Xu, Mathew Monfort, Wongun Choi, Chris Baker, Yibiao Zhao, Yizhou Wang, and Ying Nian Wu. Multi-agent tensor fusion for contextual trajectory prediction. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 12126–12134, 2019.

[69] Brian D Ziebart, Nathan Ratliff, Garratt Gallagher, Christoph Mertz, Kevin Peterson, J Andrew Bagnell, Martial Hebert, Anind K Dey, and Siddhartha Srinivasa. Planning-based prediction for pedestrians. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3931–3936, 2009.