Context models for pedestrian intention prediction by factored latent-dynamic conditional random fields

Satyajit Neogi

2020

Satyajit Neogi. (2020). Context models for pedestrian intention prediction by factored latent-dynamic conditional random fields. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/143222>

<https://doi.org/10.32657/10356/143222>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 23:33:32 SGT

**CONTEXT MODELS FOR PEDESTRIAN INTENTION
PREDICTION BY FACTORED LATENT-DYNAMIC
CONDITIONAL RANDOM FIELDS**

SATYAJIT NEOGI

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2020

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarized materials, and has not been submitted for a higher degree to any other University or Institution.

[Enter Date in dd-mm-yy]

[Provide Student's Signature Here]

..... 10-01-20
Date

..... Satyajit Neogi
[Key in Student's Name Here]

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

[Enter Date in dd-mm-yy]

.....10-01-20
Date

[Provide Supervisor's Signature Here]

..... Justin Dauwels. ...
[Key in Supervisor's Name Here]

Authorship Attribution Statement

This thesis contains material from the following 2 journal papers (1 accepted, 1 submitted) and 2 conference papers (1 published, 1 submitted) in which I am listed as an author.

1. S. Neogi, M. Hoy, W. Chaoqun, J. Dauwels, “Context Based Pedestrian Intention Prediction Using Factored Latent Dynamic Conditional Random Fields”, In Proceedings of the IEEE SSCI-2017.
2. S. Neogi, M. Hoy, K. Dang, H. Yu, J. Dauwels, “Context Model for Pedestrian Intention Prediction using Factored Latent-Dynamic Conditional Random Fields”. Accepted by IEEE Transactions on Intelligent Transportation Systems (T-ITS), available at: <https://arxiv.org/abs/1907.11881>, 2019.
3. S. Neogi, J. Dauwels, “Factored Latent-Dynamic Conditional Random Fields for Single and Multi-label Sequence Modeling”, Submitted to Journal of Machine Learning Research (JMLR), available at: <https://arxiv.org/abs/1911.03667>, 2019.
4. S. Neogi, J. Dauwels, “Factored Latent-Dynamic Conditional Random Fields for Single and Multi-label Sequence Tagging”, Submitted to IEEE ICASSP-2020.

Chapter 3 is submitted to the Journal of Machine Learning Research (JMLR) as [3]. This work is developed over the submitted conference paper [4].

The contributions of the co-authors are as follows:

- I developed FLDCRF over LDCRF and DCRFs, gathered datasets, performed experiments and wrote the manuscript.
- My supervisor Prof. Justin Dauwels offered continuous guidance for experiments and thoroughly revised the paper for improving the paper quality.

Chapters 4 and 5 are accepted by IEEE T-ITS as [2]. This work is built on our published conference paper [1].

The contributions of the co-authors are as follows:

- I thought about the idea of dynamic context for pedestrian intention prediction. Prof. Justin Dauwels advised to go ahead with the idea.
- Dr. Michael Hoy suggested to apply LDCRF as a baseline model.

- I developed the idea and mathematics of FLDCRF over LDCRF for improved prediction.
- I implemented the ideas, collected a dataset inside NTU campus and prepared the pedestrian sequences.
- Dr. Kang Dang helped to build state-of-the-art object detection and semantic segmentation modules.
- Dr. Michael Hoy and Prof. Justin Dauwels offered continuous guidance during the experiments. Dr. Michael Hoy suggested to apply models on the JAAD data.
- I wrote the manuscript. Prof. Justin Dauwels and Dr. Michael Hoy helped me to revise the paper thoroughly.
- Dr. Hang Yu reviewed the paper and offered important revisions to the proposed model.

Chapter 1 also contains some material from [2] and [3].

[Enter Date in dd-mm-yy]

[Provide Student's Signature Here]

..... 10-01-20
Date

..... Satyajit Neogi.
[Key in Student's Name Here]

Abstract

Smooth handling of pedestrian interactions is a key requirement for Autonomous Vehicles (AV) and Advanced Driver Assistance Systems (ADAS). Such systems call for early and accurate prediction of a pedestrian’s crossing/not-crossing behaviour in front of the vehicle. Existing approaches to pedestrian behaviour prediction make use of pedestrian motion, his/her location in a scene and static context variables such as traffic lights, zebra crossings etc. We stress on the necessity of early prediction for smooth operation of such systems.

We introduce the influence of vehicle interactions on pedestrian intention for this purpose. In this thesis, we show a discernible advance in prediction time aided by the inclusion of such vehicle interaction context. We apply our methods to two public datasets, viz., Daimler dataset and JAAD dataset. We also contribute two datasets towards pedestrian behaviour prediction research, viz., NTU dataset and Little India dataset and apply our methods on these datasets. While the existing best system predicts pedestrian stopping behaviour with 70% accuracy 0.38 seconds before the actual events, our system achieves such accuracy across multiple datasets at least 0.9 seconds on an average before the actual events.

We formulate the pedestrian behaviour prediction problem as a sequence labeling task. Conditional Random Fields (CRF) are frequently applied for labeling and segmenting sequence data. In the existing literature, hidden variables have been introduced in a labeled CRF structure in order to model the latent dynamics within class labels, thus improving the labeling performance. Such a model is known as Latent-Dynamic CRF (LDCRF). We propose a generalization of LDCRF, called Factored LDCRF (FLDCRF), a structure that allows multiple latent dynamics of the class labels to interact with each other. Including such latent-dynamic interactions leads to improved performance on single-label and multi-label sequence modeling tasks.

We validate our FLDCRF models on standard single-label and multi-label sequence tagging experiments across two different datasets - UCI gesture phase data and UCI opportunity data, before proceeding to apply it for pedestrian behaviour prediction. FLDCRF outperforms all state-of-the-art sequence models, i.e., CRF,

LDL-CRF, LSTM, LSTM-CRF, Factorial CRF, Coupled CRF and a multi-label LSTM model in all our experiments. In addition, FLDCRF offers easier model selection and is more consistent across validation and test data than LSTM models. FLDCRF is also much faster to train compared to LSTM, even without a GPU. FLDCRF outshines the best LSTM model by 4% in terms of F1-score on a single-label task on the UCI gesture phase data and outperforms LSTM by $\sim 2\%$ (F1-score) on average on the multi-label sequence tagging experiment on UCI opportunity data. FLDCRF models also outperform LSTM models on the pedestrian behaviour prediction task across multiple datasets.

Interacting latent dynamics in a FLDCRF can be exploited for modeling multi-agent interactions in a social environment. FLDCRF accommodates interacting discrete latent state spaces in its structure. The same idea can be extended to interacting heterogeneous (discrete and continuous) state space models.

Thesis Supervisor:

Dr. Justin Dauwels

Associate Professor

School of Electrical and Electronic Engineering

Nanyang Technological University

Acknowledgements

Pursuing a PhD is finding paths that are not known. A PhD student not only needs guidance to avoid getting lost, but also seeks freedom to discover dreamlands. I am grateful to have a supervisor who indulges in originality and quality above anything else. I want to thank my Professor, Dr. Justin Dauwels, for a journey towards excellence. He gave me the utmost freedom to explore my ideas, and made them better with his insights. From engaging discussions to thorough revisions, interacting with him has only made me get better. I continue to be inspired by him.

A sincere moment of thanks goes to Dr. Michael Hoy, for tolerating all my fuss, for always being there. Michael had been my lifeline during the tedious initial years, where I would be lost in almost all aspects of research. Be it discussing an idea or building on it, developing codes or debugging, writing a paper or revising it, Michael made my life easier throughout my PhD. He deserves a lot of credit for this thesis.

Ideas are nothing if not continuously grown by moulding and questioning. I really enjoyed working with a wonderful team who would always come up with questions and feedback on my ideas through engaging discussions. I would like to thank all my team members, Dr. Dang Kang, Selena Jiang, Liu Letao, Athul Mathew, Prakash Khunti, Dr. Weng Chaoqun and Dr. Sarat Chandra Nagaprabhu, for creating an enjoyable research environment. A special thanks to Dr. Dang Kang, he was always there for me, helping me to develop state-of-the-art modules and guiding me in general like a brother.

I am very thankful to the members of my Thesis Advisory Committee, Prof. Jianfei Cai, Prof. Ma Kai Kuang, Prof. Xavier Bresson and Prof. Junsong Yuan, for their time and valuable feedback during each assessment. Their kind advice have helped me to be on track and improve the thesis quality.

I also want to thank ST Engineering and its people, Paul, Alex, Lance, Xu Yan, Kuok Ming for their continuous support to my PhD with resources, advices and assistance. A special thanks to my lab manager Alex for managing the lab so wonderfully. I also want to thank our STE director Paul for his support and faith on me on several occasions.

I am grateful to NTU EEE for supporting my PhD. I thank the entire EEE Graduate Programme Office for their continuous effort towards well-being of PhD students and smooth conduct of the PhD degree.

I made many friends at several stages of my NTU life. They not only helped me get through tough times, but also gave me memorable moments to cherish. I want to thank my friends Mainak, Suryashish, Syed, Zhou Hui, Abhishek, Maitreyee, Yashwant, Chaitra, Pallavi, Dwaipayan and many others. A special thanks to Mainak, for teaching me so many valuable life lessons. He deserves a lot of credit for what I have become in last few years.

My life at NTU is incomplete without Cricket. I made some incredible friends and spent some of the most memorable times on the NTU Cricket Field. I thank my mates, Nandan, Subhashish, Shafquat, Shashank, Naveen, Sakcham, Moiz, Dinis Sir, Siva Sir, Alok, Rakesh and so many others for such wonderful times.

I want to thank NTU Graduate Students' Association (NTUGSA) for giving me the opportunity to lead the Sports committee. I learned so many valuable aspects of leadership while assuming this position. I also want to thank my committee members, the NTUGSA Vice-President Souravik, and the NTUGSA President Luca for their continuous support.

The NTU campus can talk to me. It is so beautiful and every inch of it builds a healthy research environment. I thank NTU, for everything. NTU will always be close to my heart.

Any amount of thanks cannot do justice to the struggles and sacrifices of my parents, just to see me grow and face the world with truth and bravery. I dedicate this thesis to my parents.

And lastly, I give my heart to my best friend, my fiancée and the love of my life, Deblina Das. At times, she has been my only hope, my only companion, the only light in the darkest times, like my moon in the night sky. My dear Pupurani, I am looking forward to sharing a lifetime with you.

Contents

Abstract	7
Acknowledgements	9
List of Figures	14
List of Tables	19
1 Introduction	23
1.1 Motivation	23
1.2 Our Contributions	27
1.3 Thesis Overview	29
1.3.1 Chapter 2: Background	30
1.3.2 Chapter 3: Factored Latent-Dynamic Conditional Random Fields (FLDCRF)	30
1.3.3 Chapter 4: Dynamic Context model for Pedestrian Intention Prediction using FLDCRF	31
1.3.4 Chapter 5: Real-life Pedestrian Intention Prediction	31
1.3.5 Chapter 6: Conclusion	32
2 Background	33
2.1 Graphical Models	33
2.1.1 Directed Graphical Models	34
2.1.2 Undirected Graphical Models	34
2.2 Sequence Labeling	35
2.3 Conditional Random Fields	36
2.3.1 Model	36
2.3.2 Training model parameters	38
2.3.3 Inference	38
2.4 Latent-Dynamic Conditional Random Fields	39

2.5	Dynamic Conditional Random Fields	42
2.6	Forward Algorithm	43
3	Factored Latent-Dynamic Conditional Random Fields (FLDCRF)	45
3.1	Motivation	45
3.2	Literature Review	50
3.3	Variants	55
3.4	Model	56
3.4.1	Definition: FLDCRF	59
3.4.2	Training	60
3.4.3	Inference	60
3.5	Datasets	61
3.5.1	UCI Gesture Phase Dataset	61
3.5.2	UCI Opportunity Dataset	62
3.6	Experimental Setup	63
3.6.1	Metrics for evaluation	63
3.6.2	Benchmarking	64
3.6.3	Model Platforms and System Specifications	68
3.7	Results	68
3.7.1	Performance Attributes	68
3.7.2	UCI Gesture Phase Dataset	70
3.7.3	UCI Opportunity Dataset	77
3.7.4	Multi-view Experiment	80
3.8	Discussion	84
3.9	Conclusion	86
4	Dynamic Context Model for Pedestrian Intention Prediction using FLDCRF	89
4.1	Challenges	90
4.1.1	Is (s)he going to cross?	90
4.1.2	Who is going to cross?	90
4.1.3	Safe ride or smooth ride?	90
4.2	Proposed Solution	92
4.2.1	The objective statement	92
4.3	Literature Review	93
4.3.1	Pedestrian-only approaches	93
4.3.2	Context based approaches	93
4.4	Proposed Intention Prediction Model	94

4.5	Nomenclature	95
4.6	Datasets	96
4.6.1	Daimler Dataset	97
4.6.2	In-house NTU Data	98
4.7	Feature Extraction	98
4.7.1	Context Features	98
4.7.2	Motion Features	101
4.8	Experimental Setup	104
4.8.1	Data for Evaluation	104
4.8.2	Features and Labels	104
4.8.3	Comparison of the Models	105
4.9	Results	106
4.9.1	Metric Definition	106
4.9.2	Daimler Dataset	107
4.9.3	NTU Dataset	107
4.10	Failed Cases and Stability Analysis	112
4.11	Conclusion	112
5	Real-life Pedestrian Intention Prediction	115
5.1	New Challenges from Real-life Data	115
5.1.1	Moving Ego-vehicle	115
5.1.2	More Variations in Pedestrian Behaviour	116
5.1.3	Variable Pedestrian Density	117
5.1.4	Data Availability	118
5.2	How we respond to the challenges?	118
5.2.1	The revised IP model	119
5.2.2	The Revised Nomenclature	119
5.2.3	A New Dataset: Little India (LI) Data	121
5.2.4	Pedestrian Interaction Model	121
5.3	Datasets	122
5.3.1	JAAD	122
5.3.2	Little India data	123
5.4	Feature Extraction	124
5.4.1	Ego-vehicle Context	124
5.5	Experimental Setup	124
5.5.1	Data for evaluation	125
5.5.2	Features and Labels	125
5.5.3	Models compared	126

5.6	Results	127
5.6.1	Metric definition	127
5.6.2	JAAD	128
5.6.3	Little India data	131
5.7	Failed Cases and Stability Analysis	133
5.7.1	JAAD data	133
5.7.2	LI data	135
5.8	Discussion	135
5.8.1	A Real-time System	136
5.8.2	Visualization of Prediction Results	136
5.9	Conclusion	136
6	Conclusions and Future work	139
6.1	Summary of Main Contributions	139
6.2	Future Research on FLDCRF	142
6.2.1	Standard and Larger sequence datasets	142
6.2.2	FLDCRF-i	142
6.2.3	CNN-FLDCRF	143
6.2.4	Semi-supervised/Unsupervised Interaction Models	144
6.2.5	Heterogeneous state-space models	147
6.3	Future Research on the IP Model	148
6.3.1	Pedestrian-pedestrian Interactions	148
6.3.2	Augment by Static Context	148
6.3.3	End-to-end Context Models for IP	148
	Our Publication(s)	149
	Bibilography	151

List of Figures

1.1	Is he/she going to cross the street? Figures taken from the JAAD dataset [23].	24
1.2	Who is going to cross the street? Figure drawn from a crowded scene in the Little India data [5].	24
1.3	(a) A Long Short-Term Memory (LSTM) cell [87]. (b) An LSTM network [88]. (b) An LSTM-CRF network [86].	26
1.4	(a) A Linear Chain CRF [42]. (b) A Latent-Dynamic CRF (LDCRF [45]). (b) The proposed Factored LDCRF [1, 2].	27
1.5	A causal model for pedestrian behaviour prediction. Two types of causal/context variables are shown: static and dynamic contexts. Figure sources: Google images, NTU dataset [1] and Little India data [5]	28
2.1	A directed graphical model.	33
2.2	An undirected graphical model.	35
2.3	A Linear Chain Conditional Random Field.	36
2.4	A Latent-Dynamic Conditional Random Field.	40
2.5	Dynamic Conditional Random Fields [44].	42
3.1	Single-label sequential CRF variants. White nodes are observed and black nodes are hidden. Grey nodes are observed only during training.	46
3.2	(a) A Hidden Markov Model. (b) A Factorial HMM with multiple cotemporal state variables.	46

- 3.3 (a) A Latent-Dynamic Conditional Random Field. (b) FLDCRF graphical model for single-label sequence prediction. Solid and dashed connections depict two different latent dynamics within class labels $\{y_t\}_{t=1:T}$. Dashed-dotted inter-layer connections capture interactions among different latent dynamics. The graph shows only two hidden layers and first-order Markov connections for transitions between state variables $\{h_{i,t}\}_{t=1:T}$ in layer i , $i = 1 : L$. More hidden layers and long-range dependencies (e.g., 2nd, 3rd order Markov connections for transitions, 1st, 2nd order Markov connections for dashed-dotted inter-layer influences, etc.) are also possible but omitted for simplicity. 47
- 3.4 Existing multi-label sequence learning models. (b) and (c) are examples of Dynamic Conditional Random Fields [44]. 49
- 3.5 Variants of FLDCRF. (a) FLDCRF graphical model for multi-label sequence prediction. Different label categories, $y_{1,t}$ and $y_{2,t}$, over input x_t are connected through their respective hidden layers, $h_{1,t}$ and $h_{2,t}$, influencing each others' latent dynamics. (b) FLDCRF graphical model for social interaction. Two different objects, $(\{x_{1,t}\}, \{y_{1,t}\})$ and $(\{x_{2,t}\}, \{y_{2,t}\})$ are shown to be interacting in the latent space through their hidden layers to effect each others' dynamics. Longer range dependencies are possible in both models but are omitted for simplicity. In fact, we add first-order Markov connections for influence links in our multi-label experiment (see Section 3.6). 49
- 3.6 Multi-view LDCRFs [74]. The features x_t in Fig. 3.3b are distributed across two variables $x_{1,t}$ and $x_{2,t}$, i.e., $x_t = \{x_{1,t}, x_{2,t}\}$ 53
- 3.7 The general structure of FLDCRF with the essential graphical constraints. Hidden variables $h_{i,t}$, $i = 1 : L$, $t = 1 : T$ can assume any dependency structure (first-order Markov, second-order Markov, cotemporal links between layers etc.) within themselves in this general FLDCRF structure. 57
- 3.8 Venn diagram illustrating the relationship between FLDCRF and major sequential CRF variants. The configuration underneath the model names describe (number of label categories, number of hidden layers, class alphabet and the sets of hidden states) in an FLDCRF-m model in order to derive it. For simplicity of description, we assume 1 hidden layer for each label category in the FLDCRF-m configuration, i.e., L hidden layers for L label categories. 58
- 3.9 Illustrating each ADL sequence in the UCI opportunity dataset [73]. . 62

3.10	Two different FLDCRF-m models considered in the multi-label experiment on the UCI opportunity data.	66
3.11	Training and inference times of different FLDCRF and LSTM models on UCI gesture phase data. Training time shown for each FLDCRF-s is until model convergence on training data, with the default criteria in Stan. Training time shown for LSTM is for 500 training epochs. . .	73
3.12	Average training time required by different FLDCRF settings per outer fold of the nested CV experiment on the UCI gesture phase data.	77
3.13	Average training time required by FLDCRF-m and LSTM-m per outer fold of UCI Opportunity data.	81
4.1	Is he/she going to cross? Figures taken from the JAAD dataset [23]. .	91
4.2	Who is going to cross? Figure drawn from a crowded scene in the Little India data [5]. A pre-trained Faster R-CNN [34] is applied to obtain the pedestrian and vehicle bounding box detections.	91
4.3	Safe ride or smooth ride? The AV trade-off [91, 92].	92
4.4	Overview of the proposed intention prediction system. Points in red highlight our main contributions. Figure taken from [1].	95
4.5	Types of pedestrian behaviours considered in this chapter. Figures drawn from the JAAD [23] dataset. EI stands for event instant (crossing, stopping etc.).	97
4.6	NTU dataset sample. Camera_1 (left) and Camera_2 (right) together capture the whole scene of interest.	98
4.7	Feature extraction: (a)-(c) depict pedestrian location context. Semantic segmentation of the scene in (a) is presented in (b). Road pixels are highlighted. (c) shows the different regions around pedestrian bounding box to capture necessary location context variables. (d) highlights pre-calibrated depth lines. Figures taken from the NTU dataset. (e) 9 points are selected symmetrically within a pedestrian bounding box during motion feature extraction.	99
4.8	Displacement in real world and its image equivalent. z_i image axis and z_R real world axis are pointed outwards w.r.t. the paper plane. y_R gives the ego-vehicle moving direction, i.e., the longitudinal direction.	102
4.9	Sign conventions on the image plane. θ_y is considered to be along +ve y axis is real world.	103

4.10	Labeling the training sequences for the Daimler and NTU data. ‘c’ denotes ‘crossing’ and ‘n’ denotes ‘not-crossing’ intention. For the Daimler data, $pred_ahead = 20$ (1.25 s at 16 fps). The NTU data follows similar labeling with $pred_ahead = 30$ (2 s at 15 fps).	105
4.11	Performance comparison between Daimler benchmark [7] and our motion based approach.	108
4.12	Performance of different systems on stopping sequences of NTU dataset.	109
4.13	Performance for stopping sequences with positive vehicle context in the NTU dataset.	110
4.14	Results on continuous crossing sequences of the NTU dataset.	111
4.15	Overall classification accuracy of different systems on the NTU dataset.	111
4.16	Individual sequence outputs by $FLDCRF - msv$ on the NTU dataset. Failed cases are shown in bold red.	112
5.1	Ego-motion compensation from optical flow. (a) Original image [37]. (b) Uncompensated optical flow [32]. (c) Ego-compensated optical flow by [33]. The normalized flow values are plotted in grayscale.	116
5.2	Examples of real-life pedestrian encounters not covered in Chapter 4.	117
5.3	Variable pedestrian density areas while navigating. Figures from LI [5] and JAAD [23] datasets.	118
5.4	The revised intention prediction system for real-life data.	119
5.5	Examples of common pedestrian sequence types observed from a vehicle (from JAAD [23] dataset). EI stands for event instant (crossing, stopping etc.).	121
5.6	FLDCRF graphical model for pedestrian-pedestrian interactions. Two different pedestrians, $(\{x_{1,t}\}, \{y_{1,t}\})$ and $(\{x_{2,t}\}, \{y_{2,t}\})$ are shown to be interacting in the latent space through their hidden layers to effect each others’ dynamics. Longer range dependencies are possible but are omitted for to avoid clutter. The vehicle context is included in the input features $x_{i,t}$ of pedestrian i . A spatial grid (b), similar to [21], is included to allow only nearby pedestrians to interact.	122
5.7	Considered JAAD pedestrian sequences in our evaluation.	123
5.8	Labeling the training sequences in the JAAD and LI data. ‘c’ denotes ‘crossing’ and ‘n’ denotes ‘not-crossing’ intention.	126
5.9	Accuracy performances of different systems vs time on the JAAD data.	129
5.10	Overall classification accuracy of different systems on JAAD sequences.	131
5.11	Comparison of different systems vs time on the LI data.	132
5.12	Overall classification accuracy of different systems on LI sequences.	133

5.13	Individual sequence outputs by $FLDCRF-(mtn+veh)$ on the JAAD dataset. Failed cases are shown in bold red.	134
5.14	Failed prediction by $FLDCRF-mv$ on standing sequence 0208_02. cp deontes the critical point ('0' mark). GT is the ground-truth intention at '0'.	135
5.15	Sample prediction outcomes by the $FLDCRF-mv$ system on the JAAD dataset [23]. A green bounding box indicates 'not-crossing', a pink bounding box indicates 'crossing' and a red bounding box indicates 'crossing within 20 meters'.	137
6.1	FLDCRF graphical model for social interaction. Two different objects, $(\{x_{1,t}\}, \{y_{1,t}\})$ and $(\{x_{2,t}\}, \{y_{2,t}\})$ are shown to be interacting in the latent space through their hidden layers to effect each others' dynamics. Longer range dependencies are possible in both models but are omitted for simplicity.	143
6.2	A CNN-FLDCRF model.	144
6.3	Semi-labeled and Unlabeled FLDCRF interaction variants.	145
6.4	A FLDCRF with heterogeneous interacting state space.	147
6.5	Realizing the context model in Fig. 1.5 by revising our models. The revisions are shown in red. 'TL' stands for traffic lights, 'ZC' for zebra crossings, and 'BS' for bus stops.	149
6.6	An end-to-end context model for pedestrian intention prediction. . . .	149

List of Tables

3.1	Number of instances in each sequence in UCI gesture phase dataset [76].	62
3.2	Number of instances in each sequence in the nested CV experiment on UCI opportunity dataset [73].	64
3.3	Information on the UCI gesture phase [76] and the UCI opportunity dataset [73].	65
3.4	Hyperparameter settings for FLDCRF-s, FLDCRF-m, LSTM and LSTM-m for experiments in this chapter. We consider same hyperparameters for LSTM and LSTM-CRF.	67
3.5	F1-scores of the models for the experiment proposed by [78] on UCI gesture phase data. The best LSTM-CRF model on the validation set achieved a F1-score 84.88 without retraining on the training+validation set.	70
3.6	FLDCRF-s validation performance on experiment 2 [78] on UCI gesture phase data.	71
3.7	LSTM-CRF validation performance on experiment 2 [78] on UCI gesture phase data. N_{els} denotes the number of training epochs and N_{hls} represents the number of hidden units in the model.	72
3.8	F1-scores on the nested CV experiment on UCI gesture phase data. .	74
3.9	FLDCRF-s validation performance on nested cross-validation sets 4 and 7 of the nested CV experiment on UCI gesture phase data. . . .	74
3.10	LSTM validation performance on nested validation sets 4 and 7 of the nested CV experiment on UCI gesture phase data. N_{els} denotes number of epochs trained and N_{hls} represents the number of hidden units in the model.	75
3.11	Summary of cross-validation performance by FLDCRF-s and LSTM on the inner loops of nested CV experiment on the UCI gesture phase data.	76
3.12	F1-scores for different models on the ‘locomotion’ activity label during nested CV experiment on UCI opportunity data.	78

3.13	F1-scores for different models on the ‘high-level’ activity (HL) label during nested CV experiment on UCI opportunity data.	79
3.14	F1-scores for different models on the joint sequence labeling task during nested CV experiment on UCI opportunity data.	80
3.15	FLDCRF-m2 validation performance on nested cross-validation sets 2 and 5 on UCI opportunity data.	81
3.16	LSTM-m validation performance on nested cross-validation sets 2 and 5 on UCI opportunity data.	82
3.17	Summary of cross-validation performance for FLDCRF-s and LSTM on the inner loops of nested CV experiment on the UCI opportunity data.	83
3.18	Comparing different models by distributing body-worn (113) and shoe sensor (32) features. LDCRF-body, LDCRF-shoe and LDCRF-(body+shoe) corresponds to $x_{1,t}$, $x_{2,t}$ and x_t respectively. LDCRF-averaged gives the mean of LDCRF-body and LDCRF-shoe outputs. MVLDCRF-averaged performance corresponds to the training and weighted inference mechanism by [74], while MVLDCRF-body layer and MVLDCRF-shoe layer gives the F1-scores obtained from individual layers of Fig. 3.6.	84
3.19	Comparison of FLDCRF and LSTM modeling attributes. ‘++’ stands for very convenient for users, ‘+’ stands for convenient, ‘-’ stands for inconvenient and ‘- -’ stands for very inconvenient.	85
4.1	Table showing distribution of sequences in Daimler dataset	97
4.2	Table showing distribution of test sequences in different datasets. . .	104
4.3	Comparison between FLDCRF, LDCRF and LSTM over nested validation sets on <i>mtn + spa + veh</i> system (on the NTU data).	109
5.1	Table showing distribution of test sequences in different datasets. . .	125
5.2	Classification accuracy of all considered JAAD standing & starting sequences by different systems within different prediction windows. .	130
5.3	Classification accuracy of all considered JAAD continuous crossing & stopping sequences by different systems within different prediction windows.	130

Chapter 1

Introduction

In this chapter, we describe the context of this thesis and its relevance to the real-world problems we address. We setup the stage with the problems at hand, and then briefly introduce our solutions. Later in the chapter, we discuss the thesis organization, how it should be read, and outline each chapter that follows.

1.1 Motivation

As¹ we enter the era of autonomous driving with the first ever self-driving taxi launched in December 2018, smooth handling of pedestrian interactions still remains a challenge. The trade-off lies between on-road pedestrian safety and smoothness of the ride. Recent user experiences and available online footage suggest conservative autonomous rides resulting from the emphasis on on-road pedestrian safety. To achieve rapid user adoption, the AVs must be able to simulate a smooth human driver-like experience without unnecessary interruptions, in addition to ensuring 100% pedestrian safety.

Automated braking in the Advanced Driver Assistance Systems (ADAS) tackle the emergency pedestrian interactions. The brakes are activated when pedestrians are detected within the vehicle safety range. Such a system may need to brake strongly when the pedestrians are detected late, which can occur frequently in crowded areas. A future ADAS must be able to offer a smoother experience on such interactions.

The key to a safe and smooth autonomous interaction with pedestrians lies in stable, early and accurate prediction of whether the pedestrian(s) in front of the vehicle intend to cross the road. Accurate and timely prediction of the pedestrian crossing behaviour ensures on-road pedestrian safety, while early anticipation of the crossing/not-crossing behaviour offers more path planning time and consequently a

¹Content taken from our papers [1, 2].

smoother control over the vehicle dynamics. Moreover, stable and early prediction of a pedestrian's crossing behaviour (see Fig. 1.1) could give assurance to human drivers, especially while riding through a crowded scene (see Fig. 1.2).

Figure 1.1: Is he/she going to cross the street? Figures taken from the JAAD dataset [23].

Recent works on on-road pedestrian behaviour prediction (see [7] - [20], described in detail in Chapter 4.3) rely on various attributes of the pedestrians, such as their motion, skeletal pose, their location in the scene (on road, at curb etc.), and certain static context variables (e.g., presence of zebra crossings, traffic lights etc.). While pedestrian motion, skeletal pose and location are reliable indicators of the current pedestrian action, they hardly reflect on the long-term (>1 sec ahead) pedestrian behaviour. Static context variables (e.g., zebra crossings, traffic lights) certainly are important factors to predict future behaviour, but are not always present in a scene.

Figure 1.2: Who is going to cross the street? Figure drawn from a crowded scene in the Little India data [5].

More often than not, an Autonomous Vehicle is likely to encounter pedestrian(s) without such contexts. Therefore, we need to devise an early pedestrian behaviour prediction model that is robust to all possible scenarios.

Since each pedestrian interaction with the ego-vehicle is continuous in time for a certain duration T_p and we want to classify crossing/not-crossing intentions of a pedestrian at each instant t within this duration, we formulate this problem as a continuous online sequence labeling task (see Chapter 2.2), $P(y_t \mid x_{1:t})$, $t = 1: T_p$, where y_t is the intention variable to be predicted and x_t are the input features to the system at t .

Long Short-Term Memory (LSTM, see Fig. 1.3a-b) networks [58] are most popular for sequence modeling, largely due to their ability to capture long-range dependencies among sequence values. They are frequently applied to sequence labeling tasks, viz., named entity recognition [48, 53], noun phrase chunking [53], action recognition [29] etc. A variant of LSTM, combined with a CRF layer [48] for classification (see Fig. 1.3c), has recently been very successful for sequence labeling [48, 53, 59]. Although LSTM (and LSTM-CRF) models have reported some outstanding performances in the recent past, these models are well known for tedious hyperparameter optimization, long training and their reliance on massive datasets. In addition to these attributes, we observe inconsistency across validation and test performance for LSTM models during our experiments (see Chapter 3, on datasets with order of 10,000 datapoints), i.e., the optimized models fail to consistently perform on the test data. Moreover, the models are not very intuitive for beginner to regular users to debug and interpret. Despite occasional outstanding performances, constraints such as long training, availability of big data, tedious model selection procedures, and most importantly uncertain performance on test data (as we demonstrate in Chapter 3) raise concerns about the stability of the LSTM models in practical deployment. And for an application as critical as an on-road pedestrian behaviour prediction, unstable and inaccurate outputs even for a short duration can prove to be fatal.

An alternative popular solution to sequence labeling is through probabilistic models, e.g., Conditional Random Fields (CRF). Such models not only offer an intuitive mathematical and graphical structure over the input and output variables, but also present easy model selection (if necessary) and consistent performance across validation and test data (see Chapter 3). The most popular and best performing CRF type model for sequence labeling is a Latent-Dynamic CRF [45], which has been shown to outperform other popular probabilistic sequence models viz., Linear Chain CRF [42], Hidden CRF (HCRF [54]) and Hidden Markov Models (HMM [46]) in labeling and segmentation tasks [45, 55]. An LDCRF (see Fig. 1.4b) models the underlying latent-

Figure 1.3: (a) A Long Short-Term Memory (LSTM) cell [87]. (b) An LSTM network [88]. (b) An LSTM-CRF network [86].

dynamics within the input ($\{x_t\}_{t=1:T}$) and output ($\{y_t\}_{t=1:T}$) sequence variables by inserting a layer of hidden variables ($\{h_t\}_{t=1:T}$) to the simple Linear Chain CRF (see Fig. 1.4a) structure. Although [82] presents balanced performance by LDCRF and LSTM with each outperforming the other on different tasks, majority of the recent papers [83, 84] show LSTM/LSTM-CRF to outperform LDCRF on sequence labeling tasks. We also observe LSTM/LSTM-CRF to outperform LDCRF on the test data during some of our nested cross-validation (CV) experiments (see Chapter 3). Despite the easier interpretation of the LDCRF models, and their consistency across validation and test performance (see Chapter 3), one may choose LSTM/LSTM-CRF over LDCRF, since they often perform better on test sets. Therefore, we need to develop models that perform well at test time, and at the same time, perform consistently across validation and test sets, provide straightforward model selection, easy interpretation, and short training time.

Motivated by these challenges, we define the following two objectives in this thesis:

1. To devise a stable and early pedestrian behaviour prediction model for AVs and ADAS, that is robust to all possible scenarios.
2. To design a machine learning model for sequence labeling and prediction that
 - is best performing on test scenarios,
 - is consistent across validation and test data,

Figure 1.4: (a) A Linear Chain CRF [42]. (b) A Latent-Dynamic CRF (LDCRF [45]). (c) The proposed Factored LDCRF [1, 2].

- provides a smooth model selection process,
- can be interpreted easily, and
- can be trained fast and on small/medium datasets.

1.2 Our Contributions

A pedestrian’s motion is the outcome of his/her intention of crossing/not-crossing in front of a vehicle (see Fig. 1.5). Thus, relying solely on pedestrian motion behaviour is likely to result in a delayed prediction of pedestrian intention. Therefore, in order to predict a pedestrian’s crossing/not-crossing behaviour early, we examine the causal factors (see Fig. 1.5) that drive the pedestrian intention. Static context variables, e.g., pedestrian location (on road, at curb etc.), traffic lights, zebra crossings, bus stops etc. have been examined previously in [8, 11, 12, 14]. However, as discussed earlier, the majority of these context variables are not always present during driving time, and hence we need causal variables that are robust across all scenarios for early prediction of pedestrian behaviour.

As pedestrians, our crossing/not-crossing intentions are primarily influenced by the distances and speeds of the approaching vehicles. We refer to this causal information as dynamic context or *vehicle interaction context* (see Fig. 4.4) and utilize it together with pedestrian motion and location information for early prediction of future behaviour. To the best of our knowledge, we are the first² [1, 3] to consider influence of such dynamic context on pedestrian intention, which proves to be the strongest factor for early prediction, as we demonstrate in Chapters 4 and 5. Adding such context improves prediction time of the pedestrian stopping (at the curb) be-

²One of the thesis examiners kindly pointed out that vehicle context has been utilized for pedestrian path prediction in [18]. We apply this context in a more generic and simplified manner dedicated for pedestrian intention prediction, that leads substantial improvement in prediction accuracy and time, as compared to [18].

haviour by ≥ 0.5 seconds on average across considered datasets, and can potentially reduce unnecessary halts in future autonomous rides. This dynamic context also improves prediction time and accuracy of other behaviour types (e.g., crossing, starting etc., see Chapter 5.2.2) across the datasets.

Figure 1.5: A causal model for pedestrian behaviour prediction. Two types of causal/context variables are shown: static and dynamic contexts. Figure sources: Google images, NTU dataset [1] and Little India data [5]

We formulate the pedestrian intention prediction problem as a continuous online sequence labeling task (see Chapter 2.2), $P(y_t | x_{1:t})$, $t = 1: T_p$, where y_t is the intention variable to be predicted and x_t are the input features to the system at t . As described earlier, existing state-of-the-art sequence labeling models (LDCRF, LSTM, LSTM-CRF) present a trade-off between several modeling aspects, viz., test performance, ease of model selection, consistency across validation and test performance, model interpretation, and computational cost (see Chapter 3 for details).

To address this existing trade-off in modeling attributes among the state-of-the-art sequence labeling models, we propose a generalization of LDCRF, called Factored LDCRF (FLDCRF [2], see Fig. 1.4c). In contrast to a standard LDCRF, an FLDCRF accommodates multiple interacting hidden layers, and thus models multiple interacting latent dynamics of the class labels. We show FLDCRF to outperform LDCRF, LSTM and LSTM-CRF on test data across experiments in this thesis (see Chapters 3 to 5). In addition, FLDCRF offers beneficial features of probabilistic models, viz., easy model selection, consistency across validation and test performance, intuitive model interpretation, and low computational complexity (see Chapter 3.7).

The interacting hidden layers in an FLDCRF also allow multi-label sequence tag-

ging (FLDCRF-m, see Fig. 3.5a) and multi-agent sequential interactions (see Fig. 3.5b). See Chapter 3 for more details on FLDCRF. We discuss how we construct the model, derive its general mathematical formulation, and present results of its different variants on two different datasets [73, 76].

Therefore, the primary contributions of this thesis are two-fold:

1. We propose a generalization of Latent-Dynamic Conditional Random Fields (LDCRF [45]), called Factored LDCRF [1, 2, 3], for single/multi-label sequence modeling. FLDCRF outperforms current state-of-the-art LDCRF, LSTM and LSTM-CRF models across experiments in Chapters 3-5. FLDCRF also offers beneficial features of probabilistic models, viz., easy model selection, consistency across validation and test performance, intuitive model interpretation, and low computational complexity (see Chapter 3.7).
2. We introduce the influence of vehicle interactions (dynamic context) on pedestrian intention for early and accurate prediction (see Chapters 4 and 5). We apply our methods on an in-house dataset [4] specifically designed to capture vehicle influences on pedestrian intention, collected inside the Nanyang Technological University campus (NTU dataset). We extend our approach to real-life datasets, viz., the public JAAD (Joint Attention for Autonomous Driving [23]) data and the self-collected Little India³ (LI [5]) data in Chapter 5. We make the NTU data and the Little India data publicly available for future research.

1.3 Thesis Overview

The remainder of this thesis is organized as follows. In Chapter 2, we provide the necessary background on probabilistic graphical models [6], Conditional Random Fields (CRF [42]), Dynamic CRFs (DCRF [44]), Latent-Dynamic CRF (LDCRF [45]), and inference mechanisms for CRF models, e.g., forward algorithm [46]. We introduce the proposed Factored Latent-Dynamic Conditional Random Fields (FLDCRF) in Chapter 3, and draw comparisons between FLDCRF and LSTM according to several modeling criteria. In Chapter 4, we present our dynamic context model with FLDCRF for pedestrian behaviour prediction. We extend this context model for application on real life datasets in Chapter 5. We compare FLDCRF and LSTM models across experiments in Chapters 4 and 5. Finally, we offer concluding remarks and ideas for future research in Chapter 6. We briefly introduce each chapter below.

³collected by a joint effort of ST Engineering and our research group.

Each chapter in this thesis can be read independently, we provide references to dependencies (on other chapters) whenever necessary.

1.3.1 Chapter 2: Background

FLDCRF is a generalization of LDCRF [45] and Dynamic CRFs (DCRF [44]). Both LDCRF and DCRFs are developed from the concept and definition of a Conditional Random Field (CRF [42]). CRFs are defined by undirected probabilistic graphical models (PGMs [6]). Thus, in order to describe the proposed FLDCRF model, we need necessary background on PGMs, CRF, DCRFs and LDCRF.

We begin this chapter by briefly discussing probabilistic graphical models (PGMs [6]) and types of PGMs. Then we describe the CRF model, and its training and inference mechanisms. We develop the concepts of LDCRF [45] and DCRFs [44] starting from CRFs and present necessary mathematical details.

We also briefly describe the forward algorithm [46], which has been utilized for training and inference in all CRF type models in this thesis.

1.3.2 Chapter 3: Factored Latent-Dynamic Conditional Random Fields (FLDCRF)

The pedestrian behaviour prediction task requires a stable, consistent and high-performing sequence labeling model. Existing state-of-the-art for sequence labeling either offer stability and consistency (LDCRF), or offer better performance (LSTM) for datasets with the order of 10,000 datapoints (see Chapter 3). In addition to unstable and inconsistent performance, LSTM models also require tedious hyperparameter optimization and intense training.

In this chapter, we propose a generalization of LDCRF, called the Factored LDCRF (FLDCRF [1, 2]). FLDCRF models contain multiple interacting latent-dynamics within class labels and enhance the performance of a standard LDCRF, outperforming popular LSTM and LSTM-CRF models on the test data. FLDCRF models also are easier to interpret, have shortened training time, require less training data, and have consistent performance across validation and test sets.

FLDCRF models have three unique variants, viz., FLDCRF-s for single-label sequence modeling, FLDCRF-m for multi-label sequence modeling, and FLDCRF-i for modeling interactions in a social environment. We mathematically show FLDCRF to subsume major sequential CRF variants, viz., LDCRF, DCRF and LCCRF.

FLDCRF-m models latent-dynamic interactions among multiple label categories, e.g., high-level action (e.g., ‘relaxing’, ‘exercising’ etc.) and low level action (e.g.,

‘sitting’, ‘lying’, ‘running’ etc.). We show FLDCRF-m to outperform all state-of-the-art single and multi-label sequence models, viz., CRF, LDCRF, LSTM, LSTM-CRF, Factorial CRF [44] and a multi-label LSTM model (LSTM-m [66]).

We present several possible future directions rooted on FLDCRF and the idea of latent sequential interactions in Chapter 6.

1.3.3 Chapter 4: Dynamic Context model for Pedestrian Intention Prediction using FLDCRF

We need stable and early pedestrian behaviour prediction models for future safe and smooth autonomous rides. We begin this chapter by describing the problem and present a simple formulation of the intention prediction problem as an online sequence labeling task (see Section 2.2 in Chapter 2). We also define different types of pedestrian behaviours.

Existing approaches ([7] – [20]) to pedestrian behaviour prediction rely on the motion, skeletal pose of the pedestrians, and on static context variables (e.g., zebra crossings, traffic lights etc.). Pedestrian motion, skeletal pose etc. follow the intention of the pedestrians and hence relying on such information may not aid to predict the intention early. Static context variables (e.g., zebra crossings, traffic lights etc., see Fig. 1.5) are useful for early prediction of intention, but are not always present during driving time.

In order to propose a robust model for pedestrian intention prediction, we introduce the influence of approaching vehicles on pedestrian intention [1, 3] of crossing/not-crossing the road. We call this causal information *vehicle interaction context*. We show such a dynamic context to significantly improve prediction time and accuracy of all considered pedestrian behaviour types.

Due to lack of pedestrian datasets with spontaneous pedestrian-vehicle interactions, we collected our own data (NTU data [4]) for experiments. We provide this dataset publicly to researchers for testing early behaviour prediction models. FLDCRF outperforms LSTM across experiments on the NTU data.

1.3.4 Chapter 5: Real-life Pedestrian Intention Prediction

NTU dataset is captured by means of stationery cameras. While it contains spontaneous pedestrian behaviour influenced by real vehicles in the scene, the data is not captured from a moving ego-vehicle, as it should be in real-life scenarios. At the same time, it only contains limited variations in pedestrian behaviour (see Chapter 4).

We extend our intention prediction (IP) models to real-life pedestrian data in this chapter. Real-life intention prediction comes with additional challenges, viz., moving ego-vehicle, more variations in pedestrian behaviour, variable pedestrian density etc. We describe these additional challenges briefly and address them with our proposed solutions. In order to account for majority of pedestrian interactions for a vehicle, we propose a standard nomenclature to encode pedestrian behaviour types of interest.

We apply our revised models on two real-life datasets, viz., Joint Attention for Autonomous Driving (JAAD [23]) and Little India data (LI [5]). We collected the Little India data ranging from sparse to densely populated areas in downtown Singapore by a joint effort together with ST Engineering. See Chapter 5 for more details on the Little India data.

We observe similar results to Chapter 4, with context models significantly improving the prediction time and accuracy across pedestrian behaviour types. FLDCRF outperforms LSTM across the datasets.

In addition to the revised IP models, we contribute the Little India dataset (to be released soon [5]) for research on pedestrian behaviour prediction.

1.3.5 Chapter 6: Conclusion

We conclude the thesis by summarizing our contributions. We also offer ideas for future research on FLDCRF models and pedestrian behaviour prediction.

Chapter 2

Background

In this Chapter, we review the necessary background material for the rest of the thesis.

2.1 Graphical Models

A probabilistic graphical model (PGM) [6] is a model that encodes the conditional independencies between random variables in a graph structure, thus reducing the number of model parameters to be learned. The variables are represented by nodes and their inter-dependencies are captured via directed or undirected links. An example is shown in Fig. 2.1, where variables X_1 , X_2 , X_3 and X_4 and their conditional dependencies are shown by a directed graphical model.

A graphical model can be either *directed* or *undirected*, as we will briefly explain in the following.

Figure 2.1: A directed graphical model.

2.1.1 Directed Graphical Models

A directed graphical model (DGM), also called a Bayesian network, is a directed acyclic graph that encodes the conditional independencies between random variables. Fig. 2.1 is a directed graphical model. Any node in the graph is said to be independent of the others if all its parents are given (observed).

Let us consider X_1 , X_2 , X_3 and X_4 in Fig. 2.1 as binary random variables. The complete joint probability distribution $P(X_1, X_2, X_3, X_4)$ requires a knowledge of $2^4 - 1 = 15$ parameters, each corresponding to one of the possible combinations of the four variables. By leveraging Bayes' rule and the conditional independencies captured by the graphical model depicted in Fig. 2.1, we can factorize the distribution as:

$$\begin{aligned} P(X_1, X_2, X_3, X_4) &= P(X_1) \cdot P(X_2 \mid X_1) \cdot P(X_3 \mid X_1, X_2) \cdot P(X_4 \mid X_1, X_2, X_3) \\ &= P(X_1) \cdot P(X_2) \cdot P(X_3 \mid X_1, X_2) \cdot P(X_4 \mid X_3), \end{aligned} \quad (2.1)$$

The factored distribution now requires knowledge of $1 + 1 + 4 + 2 = 8$ parameters. Thus we can have a reduction of 7 parameters by knowing the dependency structure between the variables in Fig. 2.1.

In this thesis we mainly consider undirected graphical models. For further reading on directed graphical models, we refer to [6].

2.1.2 Undirected Graphical Models

An undirected graphical model, also known as a Markov network or a Markov Random Field (MRF), encodes the dependencies between random variables (nodes in the graph) through undirected links. Unlike DGMs, undirected models can contain loops. Any node is said to be independent of the rest if its immediate neighbours are given (observed). Fig. 2.2 shows the equivalent undirected graphical model of Fig. 2.1 over variables X_1 , X_2 , X_3 and X_4 . The factorization of the joint distribution $P(X_1, X_2, X_3, X_4)$ using the graph structure is given by:

$$P(X_1, X_2, X_3, X_4) = \frac{1}{Z} \phi_1(X_1, X_2, X_3) \cdot \phi_2(X_3, X_4). \quad (2.2)$$

The factors that parameterize a Markov network (here $\phi_1(X_1, X_2, X_3)$ and $\phi_2(X_3, X_4)$) are known as *clique potentials*. A *clique potential* does not necessarily correspond to a probability distribution. The normalizing constant Z , often called a *partition function*, converts the potential to a probability distribution.

Directed and undirected graphical models are inter-convertible (see [6]).

Figure 2.2: An undirected graphical model.

2.2 Sequence Labeling

Labeling and segmenting sequence data is a very common problem of machine learning, which has various applications in the fields of Natural Language Processing (e.g., noun phrase chunking [44, 50], Part of Speech tagging [44, 50], named entity recognition [48, 50] etc.), computer vision (gesture recognition [45], activity recognition [49] etc.), and information extraction [52]. Given an observed input sequence $\mathbf{x} = \{x_t\}_{t=1:T}$, the sequence labeling task aims at automatically assigning labels y_t for each input value x_t . The output sequence values (y_t) usually belong to a predefined set $\mathcal{Y}$ of class labels, i.e., $y_t \in \mathcal{Y}, \forall t$. For example, in a continuous human activity recognition problem, $\mathcal{Y} = \{\text{'sitting'}, \text{'standing'}, \text{'walking'}, \text{'lying'}\}$.

Sequence labeling can be *online* or *offline*. For *online* sequence labeling, future data (x_{t+1}, x_{t+2} etc.) are not observed, i.e., the labeling task is formulated as $P(y_t \mid x_{1:t})$. By contrast, *offline* sequence labeling can utilize future data for smoothing. Keeping in mind our primary objective of real-life pedestrian intention prediction, where processing delays are not tolerable, we consider *online* sequence labeling throughout this thesis.

Approaches to sequence labeling can be broadly classified into two categories:

- Generative models (e.g., Hidden Markov Models [46]), which learn the joint distribution $P(\mathbf{y}, \mathbf{x})$ from the training data, and
- Discriminative models (e.g., Conditional Random Fields [42], Maximum Entropy Markov Models [52] etc.), which directly learn the conditional distribution $P(\mathbf{y} \mid \mathbf{x})$ from the training data.

Discriminative models have often been shown to outperform generative models on sequence labeling tasks [42, 45, 52]. In particular, Conditional Random Fields (CRF) are frequently applied to the sequence labeling problem.

2.3 Conditional Random Fields

Conditional Random Fields (CRF [42]) are very popular for sequence labeling [42, 43, 44, 45, 53, 55].

The simplest of the sequential CRF models, called a Linear Chain CRF (LCCRF, see Fig. 2.3, also called CRF), is defined over the input ($\mathbf{x} = \{x_t\}_{t=1:T}$) and output ($\mathbf{y} = \{y_t\}_{t=1:T}$) sequence variables by the graphical constraints depicted in Fig. 2.3. Due to the first-order Markov dependency assumption among the label sequence $\{y_t\}_{t=1:T}$, a LCCRF models the extrinsic dynamics (temporal dependency) within the class labels.

Figure 2.3: A Linear Chain Conditional Random Field.

2.3.1 Model

The model has two kinds of links that are repeated over time, viz., *observation* links connecting the classification label y_t to the input features x_t and *transition* links connecting y_{t-1} and y_t , which capture the first-order Markov dependency among the classification labels over time. The links are modeled by the clique potential functions $\phi_2(y_t, x_t)$ and $\phi_1(y_t, y_{t-1})$ respectively. Classification labels y_t belong to a pre-defined alphabet $\mathcal{Y}$, i.e., $y_t \in \mathcal{Y}, \forall t$. As the cliques $\phi_1(y_t, y_{t-1})$ and $\phi_2(y_t, x_t)$ are repeated over time, we obtain the joint model $P(\mathbf{x}, \mathbf{y})$ by multiplying these cliques over time t .

Thus, the conditional model $P(\mathbf{y} \mid \mathbf{x})$ of an LCCRF is given by:

$$P(\mathbf{y} \mid \mathbf{x}) = \frac{P(\mathbf{x}, \mathbf{y})}{P(\mathbf{x})} = \frac{\prod_{t=1}^T \phi_1(y_t, y_{t-1}) \phi_2(y_t, x_t)}{\sum_{\mathbf{y}} \prod_{t=1}^T \phi_1(y_t, y_{t-1}) \phi_2(y_t, x_t)}. \quad (2.3)$$

The denominator is obtained by marginalizing over $\mathbf{y}$ variables. The factors in equation (2.3), $\phi_1(y_t, y_{t-1})$ and $\phi_2(y_t, x_t)$, are usually given by exponential distributions as follows:

$$\begin{aligned}\phi_1(y_t, y_{t-1}) &= \exp \left(\sum_{i,j} \theta_{i,j} \cdot f_{i,j}(y_t = i, y_{t-1} = j) \right), \\ \phi_2(y_t, x_t) &= \exp \left(\sum_{i,l} \theta_{i,l} \cdot f_{i,l}(y_t = i, x_t = l) \right),\end{aligned}\tag{2.4}$$

where $\theta = \{\{\theta_{i,j}\}, \{\theta_{i,l}\}\}$ describe all model parameters, while $f_{i,j}$ and $f_{i,l}$ are known as feature functions. For simplicity [41], we describe all feature functions and model parameters by a single index k , which give a more generic representation of sequential CRFs:

$$P(\mathbf{y} \mid \mathbf{x}, \theta) = \frac{1}{\mathbf{Z}(\mathbf{x}, \theta)} \exp \left(\sum_{t=1}^T \sum_k \theta_k \cdot f_k(y_t, y_{t-1}, \mathbf{x}, t) \right),\tag{2.5}$$

where index k ranges over all parameters $\theta = \{\theta_k\}$ and $\mathbf{Z}(\mathbf{x}, \theta)$ is the partition function given by:

$$\mathbf{Z}(\mathbf{x}, \theta) = \sum_{\mathbf{y}} \exp \left(\sum_{t=1}^T \sum_k \theta_k \cdot f_k(y_t, y_{t-1}, \mathbf{x}, t) \right).\tag{2.6}$$

As described earlier, the feature functions $f_k(y_t, y_{t-1}, \mathbf{x}, t)$ in the simple LCCRF (see Fig. 2.3) are of two types, viz., a *state* or *observation* function $s_k(y_t, x_t)$ and a *transition* function $t_k(y_t, y_{t-1})$, which are usually defined by the following combinations of linear and indicator functions:

$$\begin{aligned}t_k(y_t, y_{t-1}) &= \mathbb{1}_{\{(y_t, y_{t-1})=k\}}, \\ s_k(y_t, x_t) &= \mathbb{1}_{\{y_t=k\}} \cdot x_t,\end{aligned}\tag{2.7}$$

assuming one-hot representations for discrete x_t components. The *state* functions can be made more generic to include past and future observations from $\mathbf{x} = \{x_t\}_{t=1:T}$.

The CRF model in equation (2.5) can be further generalized to allow higher order Markov dependencies among the class labels $\mathbf{y} = \{y_t\}_{t=1:T}$, as follows:

$$P(\mathbf{y} \mid \mathbf{x}, \theta) = \frac{1}{\mathbf{Z}(\mathbf{x}, \theta)} \exp \left(\sum_{t=1}^T \sum_k \theta_k \cdot f_k(\mathbf{y}, \mathbf{x}, t) \right),\tag{2.8}$$

where $\mathbf{Z}(\mathbf{x}, \theta)$ is given by:

$$\mathbf{Z}(\mathbf{x}, \theta) = \sum_{\mathbf{y}} \exp \left(\sum_{t=1}^T \sum_k \theta_k \cdot f_k(\mathbf{y}, \mathbf{x}, t) \right).\tag{2.9}$$

Sometimes, equation (2.8) is written more compactly by the following:

$$P(\mathbf{y} \mid \mathbf{x}, \theta) = \frac{1}{\mathbf{Z}(\mathbf{x}, \theta)} \exp \left(\sum_k \theta_k \cdot F_k(\mathbf{y}, \mathbf{x}) \right), \quad (2.10)$$

where $F_k(\mathbf{y}, \mathbf{x}) = \sum_{t=1}^T f_k(\mathbf{y}, \mathbf{x}, t)$. We will utilize this CRF formulation by equation (2.10) in subsequent LDCRF and FLDCRF model descriptions.

2.3.2 Training model parameters

The CRF model parameters $\theta = \{\theta_k\}$ are often estimated by maximizing the conditional log-likelihood of the training data:

$$\mathbf{L}(\theta) = \sum_{n=1}^N \log P(\mathbf{y}^{(n)} \mid \mathbf{x}^{(n)}, \theta) - \frac{\|\theta\|^2}{2\sigma^2}, \quad (2.11)$$

where N is the total number of available labeled sequences. An L2 regularizer, as in the second term of equation (2.11), is recommended in order to avoid overfitting the model to the training data.

The factors $P(\mathbf{y}^{(n)} \mid \mathbf{x}^{(n)}, \theta)$, $n = 1 : N$ are obtained from equations (2.5)-(2.7). The denominator $\mathbf{Z}(\mathbf{x}, \theta)$ of equation (2.5) is in the classic sum-product form of dynamic programming and is efficiently computed by the forward algorithm (see Section 2.6 [46]). We apply the default BFGS optimizer in Stan modeling language [31] to obtain the estimates $\hat{\theta} = \{\hat{\theta}_k\}$ for the CRF models in this thesis.

2.3.3 Inference

The inference task comprises of two steps:

1. Obtain probability estimates of the class labels $\hat{y}_t$ at each instant t given estimated model parameters $\hat{\theta}$ and the observed input sequence $\mathbf{x} = \{x_t\}_{t=1:T}$, i.e., obtain $P(y_t \mid \mathbf{x}, \hat{\theta}) \forall y_t \in \mathcal{Y}$.
2. Infer the class label $\hat{y}_t$ with the maximum probability.

The task is formulated as:

$$\hat{y}_t = \operatorname{argmax}_{y_t} P(y_t \mid \mathbf{x}, \hat{\theta}), \quad \forall t = 1, 2, \dots \quad (2.12)$$

Inference in an LCCRF could be performed by any of the following three algorithms as per problem requirement:

- Forward algorithm [46], where the inference objective is $P(y_t \mid x_{1:t})$. In other words, the inference must be performed *online*.
- Forward-backward algorithm [46], where the inference objective is $P(y_t \mid x_{1:T})$. In other words, the inference can be performed *offline*, after the entire sequence $x_{1:T}$ is observed.
- Viterbi $P(y_{1:t} \mid x_{1:t})$ algorithm [77]. Both *online* and *offline* inference can be performed by this algorithm, however, the class labels $y_{1:t-1}$ may keep changing with every new observation x_t , $t = 1 : T$.

In this thesis, we consider *online* inference $P(y_t \mid x_{1:t})$ in all our experiments for all models, and thus compute $P(y_t \mid x_{1:t})$ by the forward algorithm [46]. Since the forward algorithm is utilized during both training and inference, we describe the algorithm in Section 2.6.

2.4 Latent-Dynamic Conditional Random Fields

Due to the Markov dependency assumption among the label sequence $\{y_t\}_{t=1:T}$, a LC-CRF (see Fig. 2.3) models the extrinsic dynamics (temporal dependency) within the class labels. In addition to the extrinsic dynamics among the class labels, each class in $\mathcal{Y}$ has underlying intrinsic dynamics. For instance, each class ‘sitting’, ‘standing’, ‘walking’ etc. contains temporal transitions between certain hidden posture states. Morency et al. [45] introduced a layer of hidden variables $\{h_t\}_{t=1:T}$ (see Fig. 2.4) in a LCCRF structure, in order to model complete latent (both extrinsic and intrinsic) dynamics of the class labels. Each class in $\mathcal{Y}$ is associated with distinct sets of hidden states. Transitions among the states (to capture the intrinsic and extrinsic dynamics) are made possible through the layer of hidden variables $\{h_t\}_{t=1:T}$. LDCRF has been shown to outperform popular sequence models viz., Hidden Markov Models [46], LCCRF and Hidden-state CRFs [54] (see Chapter 3.2) on several sequence labeling tasks [45, 55].

We describe the LDCRF [45] mathematical model below. Our proposed FLDCRF (see Chapter 3) is developed over the LDCRF model structure and constraints.

Model

The task is to learn a probabilistic mapping between a time-series sequence of observed input features $\mathbf{x} = \{x_1, x_2, \dots, x_T\}$ and a sequence of observed classification labels $\mathbf{y} = \{y_1, y_2, \dots, y_T\}$. $y_t \in \Upsilon$, $\forall j = 1, 2, \dots, T$, where Υ is the set of classification

Figure 2.4: A Latent-Dynamic Conditional Random Field.

labels. The hidden variables $\mathbf{h} = \{h_1, h_2, \dots, h_T\}$ constitute the hidden layer. Each label $\ell \in \Upsilon$ is associated with a set of hidden states $\mathcal{H}_\ell$. $\mathcal{H}$ is the set of all possible hidden states written as $\mathcal{H} = \bigcup_\ell \mathcal{H}_\ell$. $\mathcal{H}_\ell$ are disjoint $\forall \ell \in \Upsilon$. Each h_t is restricted to belong to the set $\mathcal{H}_{y_t}$, i.e., $h_t \in \mathcal{H}_{y_t}, \forall t = 1, 2, \dots, T$. The conditional model is defined as:

$$P(\mathbf{y} \mid \mathbf{x}, \theta) = \sum_{\mathbf{h}} P(\mathbf{y} \mid \mathbf{x}, \mathbf{h}, \theta) P(\mathbf{h} \mid \mathbf{x}, \theta). \quad (2.13)$$

Equation (2.13) can be re-written from the graph structure in Fig. 3.1c as:

$$\begin{aligned} P(\mathbf{y} \mid \mathbf{x}, \theta) = & \sum_{\mathbf{h}: \forall h_t \in \mathcal{H}_{y_t}} P(\mathbf{y} \mid \mathbf{h}, \theta) P(\mathbf{h} \mid \mathbf{x}, \theta) \\ & + \sum_{\mathbf{h}: \exists h_t \notin \mathcal{H}_{y_t}} P(\mathbf{y} \mid \mathbf{h}, \theta) P(\mathbf{h} \mid \mathbf{x}, \theta). \end{aligned} \quad (2.14)$$

Applying model constraints, we can write the following:

$$P(y_t = \ell \mid h_t) = \begin{cases} 1, & h_t \in \mathcal{H}_{y_t=\ell} \\ 0, & h_t \notin \mathcal{H}_{y_t=\ell}. \end{cases} \quad (2.15)$$

The model in equation (2.14) can be simplified by equation (2.15) as:

$$P(\mathbf{y} \mid \mathbf{x}, \theta) = \sum_{\mathbf{h}: \forall h_t \in \mathcal{H}_{y_t}} P(\mathbf{h} \mid \mathbf{x}, \theta). \quad (2.16)$$

$P(\mathbf{h} \mid \mathbf{x}, \theta)$ is described from the Conditional Random Field formulation given by:

$$P(\mathbf{h} \mid \mathbf{x}, \theta) = \frac{1}{\mathbf{Z}(\mathbf{x}, \theta)} \exp \left(\sum_k \theta_k \cdot F_k(\mathbf{h}, \mathbf{x}) \right), \quad (2.17)$$

where index k ranges over all parameters $\theta = \{\theta_k\}$ and $\mathbf{Z}(\mathbf{x}, \theta)$ is the partition function defined as:

$$\mathbf{Z}(\mathbf{x}, \theta) = \sum_{\mathbf{h}} \exp \left(\sum_k \theta_k \cdot F_k(\mathbf{h}, \mathbf{x}) \right). \quad (2.18)$$

The feature functions F_k 's are given by:

$$F_k(\mathbf{h}, \mathbf{x}) = \sum_{t=1}^T f_k(h_{t-1}, h_t, \mathbf{x}, t),$$

Feature functions $f_k(h_{t-1}, h_t, \mathbf{x}, t)$ can be either an *observation* (also called *state*) function $s_k(h_t, \mathbf{x}, t)$ or a *transition* function $t_k(h_{t-1}, h_t, \mathbf{x}, t)$ and are defined by equation (2.7).

Training Model Parameters

Parameters of the model can be estimated by maximizing the conditional log-likelihood of the training data given by:

$$\mathbf{L}(\theta) = \sum_{n=1}^N \log P(\mathbf{y}^{(n)} \mid \mathbf{x}^{(n)}, \theta) - \frac{\|\theta\|^2}{2\sigma^2}, \quad (2.19)$$

where N is the total number of available labeled sequences. The second term in equation (2.19) is an L2-regularizer to reduce overfitting, similar to Section 2.3.2.

Inference

Given a new test sequence $\mathbf{x}$, the inference task is given by equation (2.12). Applying the LDCRF model constraints, we can re-write the inference task as:

$$\begin{aligned} \hat{y}_t &= \operatorname{argmax}_{y_t} P(y_t \mid \mathbf{x}, \hat{\theta}) \\ &= \operatorname{argmax}_{y_t} \sum_{h_t \in \mathcal{H}_{y_t}} P(h_t \mid \mathbf{x}, \hat{\theta}), \quad \forall t = 1, 2, \dots \end{aligned} \quad (2.20)$$

Since the intention prediction task must be performed online, the inference task is reduced to $P(y_t \mid x_{1:t}, \theta)$. At each time instant t , we compute the marginals $P(h_t \mid x_{1:t}, \theta)$ by the forward algorithm (see Section 2.6) and sum them according to the disjoint sets of hidden states to obtain $P(y_t \mid x_{1:t}, \theta) = \sum_{h_t \in \mathcal{H}_{y_t}} P(h_t \mid x_{1:t}, \theta)$, $t = 1, 2, \dots$. Next we infer the label y_t corresponding to the maximum probability. As mentioned in the case of an LCCRF, the forward-backward algorithm [46] and Viterbi algorithm [77] can also be applied for problems where online inference is not required.

2.5 Dynamic Conditional Random Fields

Multi-task sequence learning [44, 50] is the task to jointly tag sequence values with multiple label categories. Such learning is often helpful where different label categories are related to each other. For example, let us consider the two tasks of continuously recognizing a high-level action (e.g., with two classes, ‘relaxing’ and ‘exercising’) and a low-level action (e.g., with four classes, ‘lying’, ‘sitting’, ‘standing’ and ‘walking’) over some given input (sensor) sequence data. Very clearly the two action types are related and a joint modeling of both is likely to improve the individual recognition performances with the help of additional contextual information (see Section 3.7).

The CRF and LDCRF allow only one label variable (y_t) in their structures at each instant. To tackle a multi-label sequence tagging task with an CRF or LDCRF would involve combining different label categories $y_{1,t}$, $y_{2,t}$ etc. into a single variable $y_t = \{y_{1,t}, y_{2,t}, \dots\}$. Such CRF structures (known as coupled CRF in [67]) have been exploited in [67, 68]. However, the number of model parameters grow rapidly in these models with increasing number of classes for each label category, thereby requiring large training data. Additionally, models with such a combined state space require all label categories $y_{1,t}$, $y_{2,t}$ to form the joint variable y_t for training at all t and thus are not flexible for data with missing labels.

A solution to learn multiple interacting label categories over inputs x_t was proposed by Sutton et al. [44] (see Fig. 2.5). The structures are known as Dynamic CRFs (DCRF). Such Dynamic CRFs are very popular [44, 75] for joint modeling of multiple label sequences. Different label categories $y_{1,t}$, $y_{2,t}$ etc. interact with each other through inter-layer links. The structure in Fig. 2.5a is popularly known as a Factorial CRF.

Figure 2.5: Dynamic Conditional Random Fields [44].

These factored representations from a DCRF require fewer parameters than modeling with the joint state variable for modeling the joint conditional distribution

$P(\{\mathbf{y}_i\}_{1:L} \mid \mathbf{x}, \theta)$, and allow easy addition/removal of links depending on required model complexity and available training data. For instance, the coupled CRF model (see Fig. 3.4a) by [67] can be efficiently represented by Fig. 2.5b, with substantially reduced model parameters for the same training labels $\{y_{1,t}\}_{t=1:T}$ and $\{y_{2,t}\}_{t=1:T}$. Sutton et al. [44] showed improvements in labeling accuracy with a FCRF while considering a joint noun phrase (NP) chunking and part of speech (POS) tagging task. They also reported better performance for FCRF compared to two separate CRFs for reduced training data.

We discuss DCRFs and existing literature on multi-label sequence modeling in more detail in Chapter 3. We show how the proposed FLDCRF generalizes DCRFs with latent variables in Chapter 3.4.

2.6 Forward Algorithm

The task is to compute $P(y_t \mid x_{1:t}, \theta), \forall y_t$. Since we need to predict probabilities of y_t given the same inputs $x_{1:t}$, the task can be simplified as:

$$\begin{aligned} P(y_t \mid x_{1:t}) &\propto P(y_t, x_{1:t}) \\ &= \alpha_t(y_t). \end{aligned} \tag{2.21}$$

The $\alpha_t(y_t)$ are known as forward variables, which are computed by the following recursions given model parameters θ :

$$\begin{aligned} \alpha_t(y_t) &= \sum_{y_{t-1}} P(y_t, y_{t-1}, x_{1:t}) \\ &= \sum_{y_{t-1}} P(x_t \mid y_t) \cdot P(y_t \mid y_{t-1}) \cdot P(y_{t-1}, x_{1:t-1}) \\ &= \sum_{y_{t-1}} P(x_t \mid y_t) \cdot P(y_t \mid y_{t-1}) \cdot \alpha_{t-1}(y_{t-1}) \\ &= \frac{1}{Z} \sum_{y_{t-1}} \phi_1(y_{t-1}, y_t) \cdot \phi_2(x_t, y_t) \cdot \alpha_{t-1}(y_{t-1}), \end{aligned} \tag{2.22}$$

with $\alpha_0(y_0) = 1, \quad \forall y_0$. y_0 is a fixed initial state for CRF.

Chapter 3

Factored Latent-Dynamic Conditional Random Fields (FLDCRF)

3.1 Motivation

Labeling and segmenting sequence data is a very common problem of machine learning, which has various applications in the fields of Natural Language Processing (e.g., noun phrase chunking [44, 50], Part of Speech tagging [44, 50], named entity recognition [48, 50] etc.), computer vision (gesture recognition [45], activity recognition [49] etc.), and information extraction [52]. Given an observed input sequence $\mathbf{x} = \{x_t\}_{t=1:T}$, the labeling task aims at automatically assigning labels y_t for each input value x_t .

Conditional Random Fields (CRF [42]) are frequently applied to the sequence labeling task. The simplest of its kind is the Linear Chain Conditional Random Field (LCCRF, also called a CRF), which is defined over the input ($\{x_t\}_{t=1:T}$) and output ($\{y_t\}_{t=1:T}$) sequence variables by the graphical constraints depicted in Fig. 3.1a. The output sequence values (y_t) usually belong to a predefined set $\mathcal{Y}$ of class labels. For example, in a continuous human activity recognition problem, $\mathcal{Y} = \{\text{'sitting'}, \text{'standing'}, \text{'walking'}, \text{'lying'}\}$. Due to the Markov dependency assumption among the label sequence $\{y_t\}_{t=1:T}$, a LCCRF models the extrinsic dynamics (temporal dependency) within the class labels.

In addition to the extrinsic dynamics among the class labels, each class in $\mathcal{Y}$ has underlying intrinsic dynamics. For instance, each class ‘sitting’, ‘standing’, ‘walking’ etc. contains temporal transitions between certain hidden posture states. Morency et al. [45] introduced a layer of hidden variables $\{h_t\}_{t=1:T}$ (see Fig. 3.1c) in a LCCRF

structure, in order to model complete latent (both extrinsic and intrinsic) dynamics of the class labels. Each class in $\mathcal{Y}$ is associated with distinct sets of hidden states. Transitions among the states (to capture the intrinsic and extrinsic dynamics) are made possible through the layer of hidden variables $\{h_t\}_{t=1:T}$. LDCRF has been shown to outperform popular sequence models viz., Hidden Markov Models [46], LCCRF and Hidden-state CRFs (HCRF, [54], see Section 3.2) on several sequence labeling tasks [45, 55].

Figure 3.1: Single-label sequential CRF variants. White nodes are observed and black nodes are hidden. Grey nodes are observed only during training.

Learning performance in a LDCRF can only be improved by varying the number N_s of states associated to each class label. This only mode of variation restricts the model capabilities, and results in a rapidly growing state transition matrix with size $(N_l \cdot N_s) \times (N_l \cdot N_s)$, where $N_l = |\mathcal{Y}|$. Such increment results in greater model complexity and requires more training data. A solution for a Hidden Markov Model (HMM) with a similar structure of the state space was proposed in [56], where multiple cotemporal state variables ($h_{1,t}$, $h_{2,t}$ etc.) replaced the single hidden state variable h_t at slice t (see Fig. 3.2). The hidden states are distributed among the state variables ($h_{1,t}$, $h_{2,t}$ etc.) and all state variables assumed independent temporal dynamics over the input observations $\{x_t\}_{t=1:T}$, resulting a reduction in number of state transition parameters.

Figure 3.2: (a) A Hidden Markov Model. (b) A Factorial HMM with multiple cotemporal state variables.

In addition, it is possible that there exists multiple interacting latent dynamics

within the class labels. For example, in a continuous human activity recognition problem (e.g., with four classes, ‘lying’, ‘sitting’, ‘standing’ and ‘walking’), a human being has two interacting dynamics - one along the plane and another in the vertical direction. Such interaction among different latent dynamics can be captured by connecting state variables ($h_{1,t}$, $h_{2,t}$ etc.) across layers (see Fig. 3.3b). LDCRF (see Fig. 3.3a) ignores any possible interaction among the associated hidden states at slice t , and thus is unable to capture such interactions. Considering all these limitations of LDCRF for a single label sequence modeling and inspired by Factorial HMM, we present a generalization of LDCRF, called Factored Latent-Dynamic Conditional Random Fields (FLDCRF [3]), in order to:

- generate new models by varying the number of hidden layers,
- generate factorized models [56] with fewer parameters, which need less data for training, and
- model multiple interacting latent dynamics within class labels.

Figure 3.3: (a) A Latent-Dynamic Conditional Random Field. (b) FLDCRF graphical model for single-label sequence prediction. Solid and dashed connections depict two different latent dynamics within class labels $\{y_t\}_{t=1:T}$. Dashed-dotted inter-layer connections capture interactions among different latent dynamics. The graph shows only two hidden layers and first-order Markov connections for transitions between state variables $\{h_{i,t}\}_{t=1:T}$ in layer i , $i = 1 : L$. More hidden layers and long-range dependencies (e.g., 2nd, 3rd order Markov connections for transitions, 1st, 2nd order Markov connections for dashed-dotted inter-layer influences, etc.) are also possible but omitted for simplicity.

We denote the FLDCRF single-label sequence model (see Fig. 3.3b) as FLDCRF-s. FLDCRF-s outperforms LDCRF on UCI gesture phase data [76] and UCI opportunity data [73] (see Section 3.7).

LSTMs [58] are the most popular kind of Recurrent Neural Networks (RNN) for sequence modeling, largely due to their ability to capture long-range dependencies

among sequence values. They are frequently applied to sequence labeling tasks, viz., named entity recognition [48, 53], noun phrase chunking [53], action recognition [29] etc. A variant of LSTM, combined with a CRF layer [48] for classification, has recently been very successful in sequence labeling tasks [48, 53, 59]. Despite some outstanding reported performances, LSTM models are well known for tedious hyperparameter optimization, long training and the necessity of massive training datasets.

We compare FLDCRF-s with both LSTM and LSTM-CRF on the test data across experiments in this thesis. In addition, we analyze several important modeling aspects of FLDCRF and LSTM, viz., ease of model selection, consistency across validation and test performance, computation times for training and inference etc. FLDCRF-s not only outperforms LSTM and LSTM-CRF on test data across the experiments, but also offers easier model selection and more consistent performance across validation and test data (see Section 3.7). In addition, FLDCRF-s requires notably less training and inference times than LSTM models, even without GPU implementation.

Multi-task sequence learning [44, 50] is the task to jointly tag sequence values with multiple label categories. Such learning is often helpful where different label categories are related to each other. For example, let us consider the two tasks of continuously recognizing a high-level action (e.g., with two classes, ‘relaxing’ and ‘exercising’) and a low-level action (e.g., with four classes, ‘lying’, ‘sitting’, ‘standing’ and ‘walking’) over some given input (sensor) sequence data. Very clearly the two action types are related and a joint modeling of both is likely to improve the individual recognition performances with the help of additional contextual information (see Section 3.7).

Dynamic CRFs [44] (see Fig. 3.4b-c) are very popular [44, 75] for joint modeling of multiple label sequences. Different label categories $y_{1,t}$, $y_{2,t}$ etc. interact with each other through inter-layer links. The structure in Fig. 3.4b is popularly known as a Factorial CRF.

Similar to a Linear chain CRF, a DCRF does not contain any hidden state variables and therefore is unable to capture the latent (intrinsic and extrinsic) dynamics within the classes of different label categories (e.g., high and low-level actions). Moreover, connections between the latent dynamics of different label categories can potentially capture deeper understanding of their interactions. Considering such limitation of DCRF, we propose a multi-label variant of FLDCRF (FLDCRF-m, see Fig. 3.5a), in order to:

- introduce latent variables in a general DCRF structure,

Figure 3.4: Existing multi-label sequence learning models. (b) and (c) are examples of Dynamic Conditional Random Fields [44].

- model interactions among the latent dynamics of two or more inter-related label categories, and
- provide a general representation of DCRFs with embedded hidden state variables for each output variable. The original DCRF structure can be recovered by associating one hidden state per class of the output variables $y_{i,t}$ (see Section 3.4 for mathematical details).

Figure 3.5: Variants of FLDCRF. (a) FLDCRF graphical model for multi-label sequence prediction. Different label categories, $y_{1,t}$ and $y_{2,t}$, over input x_t are connected through their respective hidden layers, $h_{1,t}$ and $h_{2,t}$, influencing each others' latent dynamics. (b) FLDCRF graphical model for social interaction. Two different objects, $(\{x_{1,t}\}, \{y_{1,t}\})$ and $(\{x_{2,t}\}, \{y_{2,t}\})$ are shown to be interacting in the latent space through their hidden layers to effect each others' dynamics. Longer range dependencies are possible in both models but are omitted for simplicity. In fact, we add first-order Markov connections for influence links in our multi-label experiment (see Section 3.6).

A less popular CRF variant for joint sequence tagging is coupled CRF (CCRF [67], see Fig. 3.4a), that utilizes a combined state space for different label categories $y_{1,t}$, $y_{2,t}$ etc. Recurrent Neural Networks [66] have also been applied to joint tagging of sequences. We present a structured and detailed review of literature on joint sequence labeling in Section 3.2.

FLDCRF-m outperforms Factorial CRF and coupled CRFs on the multi-task sequence labeling experiment on UCI opportunity dataset [73] (see Section 3.7). FLDCRF-m also outshines all tested single-label sequence models (trained and inferred separately on different label categories), viz., CRF, LDCRF, FLDCRF-s, LSTM and LSTM-CRF on both tasks combined, while also improving the individual labeling performances on most cases. We also compare FLDCRF-m with a multi-task LSTM model, which we refer to as LSTM-m in the paper, similar to the BiRNN model by Pahuja et al. [66].

We also propose another variant of FLDCRF, called the interaction variant or FLDCRF-i (see Fig. 3.5b), which captures relationship among the dynamics of multiple agents in a social environment. FLDCRF-i can be applied to multi-agent sequence modeling and prediction tasks in a social environment, e.g., joint intention prediction of multiple pedestrians [1]. See Chapter 6.2.2 for more details on the FLDCRF-i model. We present a general mathematical description of all the 3 FLDCRF model variants (FLDCRF-s, FLDCRF-m and FLDCRF-i) in Section 3.4.

3.2 Literature Review

We discuss here existing literature on single and multi-label sequence modeling.

Approaches to sequence labeling can be broadly classified into two categories: a) Generative (e.g., Hidden Markov Models [46]), which learn the joint distribution $P(\mathbf{y}, \mathbf{x})$ from the training data; and b) Discriminative (e.g., Conditional Random Fields [42], Maximum Entropy Markov Models [52] etc.), which directly learn the conditional distribution $P(\mathbf{y} \mid \mathbf{x})$ from the training data. Discriminative models have often been shown to outperform generative models on sequence labeling tasks [42, 45, 52]. In particular, Conditional Random Fields (CRF) are frequently applied to the sequence labeling problem [48, 50, 53].

A simple CRF model, also called Linear Chain CRF (LCCRF, see Fig. 3.1a), is defined directly over the input $(\{x_t\}_{t=1:T})$ and output $(\{y_t\}_{t=1:T})$ sequence variables. The class labels within a sequence $(\{y_t\}_{t=1:T})$ have an extrinsic dynamics (temporal dependency), as well as there exists underlying intrinsic dynamics within each class. For example, in a continuous activity recognition problem with classes ‘sitting’, ‘standing’, ‘walking’ etc., each class contains temporal transitions between certain posture states. Since such posture states are hidden, the intrinsic dynamics within a particular class can be captured by associating hidden (latent) states and allowing their transitions. The simple CRF model captures the extrinsic dynamics among the class labels through its transition links (connecting y_{t-1} and y_t). However,

it does not have any latent variables in its structure and thus is unable to capture the intrinsic dynamics within the class labels.

To address this issue, Hidden-state Conditional Random Fields (HCRF [54]) introduced hidden variables in a labeled CRF structure (see Fig. 3.1b), in order to capture the intrinsic dynamics within the class labels. The hidden variables ($\{h_t\}_{t=t_1:t_2}$) can assume values from a predefined set of hidden states (usually specified by numbers) and the Markov dependency among the variables allows to capture the intrinsic dynamics within the class label $y_{t_1:t_2}$. However, such a structure requires prior segmentation of the training sequences according to the class labels and models each class label separately (similar to unsupervised HMMs for each class label, only trained by a discriminative approach). Hence, HCRF fails to capture the extrinsic dynamics between the class labels.

An alternative approach, named Latent-Dynamic Conditional Random Fields (LDCRF), was proposed by Morency et al. [45] (see Fig. 3.1c). In a LDCRF, prior segmentation of the training sequences is not necessary. Thus, LDCRF is able to capture both (latent) intrinsic and extrinsic dynamics of the class labels. LDCRF restricts the states associated to each class label to be disjoint. Each hidden state variable h_t is also constrained to belong to the states associated to the class label y_t . These two constraints help to keep the model computations (during training and inference) tractable, despite the insertion of hidden variables. LDCRF has been shown to outperform HMM, CRF and HCRF on several sequence labeling tasks [45, 55]. Thai et al. [57] recently proposed a very similar model called the Embedded-State Latent CRFs.

We propose FLDCRF-s in order to model multiple interacting latent dynamics within the class labels (see Section 3.1). We describe how FLDCRF-s mathematically generalizes a LDCRF in Section 3.4. We demonstrate improvement in performance by FLDCRF-s over LDCRF across nested cross-validation experiments on two different datasets (see Section 3.7).

LSTMs [58] are the most popular kind of Recurrent Neural Networks (RNN) for sequence modeling. They are frequently applied to sequence labeling tasks, e.g., named entity recognition [48, 53], noun phrase chunking [53], action recognition [29] etc. LSTM-CRF, a variant of LSTM, combined with a CRF layer [48] for classification, has recently been very popular for sequence labeling [48, 53, 59].

A few studies [82, 83, 84] have compared LDCRF and LSTM on the same classification tasks. While [83, 84] show LSTM and its variants to outperform LDCRF, [82] present results with each outperforming the other on different tasks. We show that although LSTM and LSTM-CRF outperform LDCRF on certain experiments,

FLDCRF-s outperforms both LSTM and LSTM-CRF across all our experiments. By contrast, the best LSTM (and LSTM-CRF) models outshine the best FLDCRF-s performance on most of the validation sets across experiments in the paper. However, the same LSTM models (with optimized hyperparameters), kept unchanged or retrained on the validation+train data, fail to beat the optimized FLDCRF-s models on the test data across experiments (see Section 3.7). In addition to this inconsistent validation and test performance, LSTM models exhibit large variations in validation performance across different hyperparameter settings, with quite a few models producing significantly lesser validation performance. This variation exists without any easily discernible patterns among different hyperparameter settings. The variable performance across validation sets, lower worst cases, and inconsistency in validation and test performance make it quite tedious to select appropriate LSTM models, and raise concerns about the stability of the models in practical deployment. On the other hand, FLDCRF with its intuitive structure, brings consistency in validation and test performance and offers a much easier model selection process. LSTM models also take significantly longer to train than FLDCRF-s models. We present results to support all the above statements in Section 3.7.

A factored variant of LDCRF, called multi-view LDCRF (MVLDCRF [74], see Fig. 3.6), was proposed to capture interactions among different latent dynamics arising due to different kinds of input features within same class labels. A MVLDCRF is a special case of FLDCRF-s, where different latent dynamics within class labels $\{y_t\}_{t=1:T}$ are learned from disjoint feature subsets $\{x_{1,t}\}_{t=1:T}$ and $\{x_{2,t}\}_{t=1:T}$ of $x_t = \{x_{1,t}, x_{2,t}\}$. Song et al. [74] showed a MVLDCRF to outperform a LDCRF trained with the entire feature set x_t in their experiments. However, such explicit distribution of features in order to force the model to learn different latent dynamics does not guarantee improved performance, as we demonstrate by our experiments (see Section 3.7.4). In some cases, performance of such a MVLDCRF falls in between those of the individual LDCRFs, i.e., one trained with $x_{1,t}$ and another with $x_{2,t}$. These results are not reported by [74]. In most other cases, a LDCRF trained with the entire feature set x_t outperforms MVLDCRF and individual LDCRFs. By contrast, a FLDCRF-s with all hidden layers modeling their dynamics from the entire x_t lets the model to learn different latent dynamics that may exist, and in general improves the LDCRF (with x_t) performance. In the case a feature distribution ($x_{1,t}$, $x_{2,t}$ etc.) is available, it is advisable to obtain individual FLDCRF-s performances on the individual feature subsets, as well as on entire x_t , and select the best performance. It is better to avoid learning from distributed features on the same training labels as in an MVLDCRF, as we demonstrate on 10 different test sets in Section 3.7.4.

Figure 3.6: Multi-view LDCRFs [74]. The features x_t in Fig. 3.3b are distributed across two variables $x_{1,t}$ and $x_{2,t}$, i.e., $x_t = \{x_{1,t}, x_{2,t}\}$.

Multi-task learning [60] is a very common problem of machine learning, where multiple label categories are trained and/or inferred jointly over certain input features. Modeling multiple labels together serves additional context information to individual labeling tasks, which help to improve the overall labeling performance. However, the model must allow appropriate label interactions for beneficial results. Multi-task learning is frequently applied to tag images with multiple labels [61, 62].

Multi-task sequence learning [44, 50] is the task to jointly tag sequence values with multiple label categories. Existing multi-task sequence labeling approaches can be broadly classified into two categories:

1. **Using a joint state variable** (see Fig. 3.4a), i.e., $y_t = \{y_{1,t}, y_{2,t}\}$ [66, 67, 68]. Li et al. [67] presented a coupled CRF (CCRF) model that allowed the two (or more) label categories to interact temporally through their coupled transition matrix. Luo et al. [68] proposed a model called JERL (Joint Entity Recognition and Linking) for joint named entity recognition and disambiguation, which is similar to a coupled CRF. A correlated BiRNN model was introduced by [66], where multiple label categories (punctuation and capitalization) were combined to pass through the same RNN hidden layer, thus capturing the multi-label interactions through the temporal transitions of the hidden layer. Although all such joint models showed improvements over the individual models, the joint state space makes the model too large with multiple classes (or latent states) for each task.
2. **Using separate state variables** ($y_{1,t}$, $y_{2,t}$ etc.) **with cotemporal links** (see Fig. 3.4b-c [44]) to capture interaction. These structures are derived from Dynamic CRFs proposed by [44]. The structure in Fig. 3.4b is also well known as a Factorial CRF (FCRF). These factored representations from a DCRF require fewer parameters than modeling with the joint state variable, and allow

easy addition/removal of links depending on required model complexity and available training data. For instance, the coupled CRF model by [67] can be efficiently represented by Fig. 3.4c, with substantially reduced model parameters for the same training labels $\{y_{1,t}\}_{t=1:T}$ and $\{y_{2,t}\}_{t=1:T}$. Sutton et al. [44] showed improvements in labeling accuracy with a FCRF while considering a joint noun phrase (NP) chunking and part of speech (POS) tagging task. They also reported better performance for FCRF compared to two separate CRFs for reduced training data.

Several other studies [50, 64, 65, 69, 70] have been conducted on joint sequence labeling. Shi et al. [70] proposed a dual layer CRF for joint decoding of different tasks (segmentation and POS tagging), however, showed marginal improvement over individual labeling with their approach. Dredze et al. [69] designed a Multi-CRF with a shared entropy based training likelihood function. Collobert et al. [50] and Changpinyo et al. [65] reviewed existing approaches to multi-task sequence learning and argued that joint learning and decoding may not necessarily give better results than individual approaches. Collobert et al. [50] also questioned the availability of fully labeled datasets as a bottleneck of such approaches. As discussed earlier, models represented by distributed state variables (see Fig. 3.4b-c) are quite flexible and can be trained on sequences with missing labels by simply adding/dropping links whenever necessary, depending on the variable (label node) availability. It is hard to tackle such data for models represented by the joint state variable (see Fig. 3.4a), as it requires both label categories $y_{1,t}$, $y_{2,t}$ to form the joint variable y_t for training at all t . While most multi-task sequence learning models have been designed for problems in Natural Language Processing (NLP), Bruce et al. [64] presented a joint model to improve the performance of action recognition and pose estimation from video data.

As elaborated in Section 3.1, we propose FLDCRF-m in order to model latent-dynamic interactions among different label categories. We describe how FLDCRF-m gives a generic expression for DCRFs with embedded hidden-state variables in Section 3.4. FLDCRF-m outperforms all state-of-the-art single (CRF, LDCRF, LSTM, LSTM-CRF) and multi-label (FCRF, CCRF, LSTM-m) models in the multi-label sequence tagging experiment.

Most studies on single and joint sequence labeling have been conducted in the context of NLP, viz., CoNLL 2000 chunking [71], CoNLL 2003 named entity tagging [72] etc. However, recent reported (F1-score) performance results on these tasks are largely driven by the quality of the input features, while the models do not show much variance in performance. By contrast, we wish to determine the effectiveness of the proposed models and therefore do not consider such datasets for evaluating

our models. We plan to apply FLDCRF on NLP tasks in the future.

We apply the proposed FLDCRF-s and FLDCRF-m models to two problems:

1. Continuous gesture recognition on the UCI gesture phase segmentation dataset [76]. The task concerns online segmentation of gestures from rest positions.
2. Continuous multi-action recognition on the UCI opportunity dataset [73]. This task aims at continuous joint recognition of two action types (locomotion and a high-level activity, see Section 3.5).

We refer to Sections 3.5 and 3.6 for a detailed description of the datasets and experiments respectively. All experiments in this thesis consider continuous online recognition problems (see Section 2.2), i.e., the inference tasks are formulated in terms of $P(y_t \mid x_{1:t})$. We do not utilize future input features (x_{t+1} , x_{t+2} etc.) and future label/state variables during modeling/testing in any of the models.

3.3 Variants

As described in Section 3.1, a fully¹ labeled FLDCRF has three variants:

1. **FLDCRF-s**: FLDCRF model for single label sequence tagging (see Fig. 3.3b). FLDCRF-s accommodates multiple interacting latent dynamics of the class labels y_t in its structure, and extends the capability of an LDCRF [45].
2. **FLDCRF-m**: FLDCRF variant for multi label sequence modeling (see Fig. 3.5a). This model captures latent dynamic interactions among different label categories $y_{1,t}$, $y_{2,t}$ etc. over given input data x_t . The latent dynamic interactions model deeper understanding of the relationship between different label categories, thereby improving the labeling performance (see Section 3.7).
3. **FLDCRF-i**: FLDCRF variant for multi agent interaction modeling (see Fig. 3.5a) in a social environment, e.g., a multi-robot collaborative environment, multiple interacting pedestrians and vehicles in a scene etc. This model captures sequential interactions among the latent dynamics of different agents in a social environment, thereby modeling their temporal relationships. See Chapter 6.2.2 for more details on the FLDCRF-i model.

All three models can be expressed via a similar mathematical representation. Since the FLDCRF-s and FLDCRF-i are special cases of the FLDCRF-m model, we describe the FLDCRF-m model below for better generalization.

¹FLDCRF can be extended for semi-labeled and unlabeled interaction models, see Chapter 6.2.4.

3.4 Model

We will now explain how we construct the FLDCRF model, give its generic expression and then introduce the simplified versions we apply in the experiments.

Fig. 3.5a shows the graph structure for FLDCRF in multi-label classification problems (FLDCRF-m). We depict the model for $L = 2$ different label categories $\{y_{1,t}\}_{t=1:T}$ and $\{y_{2,t}\}_{t=1:T}$, interacting with each other through their respective hidden layers $\{h_{1,t}\}_{t=1:T}$ and $\{h_{2,t}\}_{t=1:T}$. In the case of a FLDCRF single-label model (FLDCRF-s, see Fig. 3.3b), the label category $\{y_t\}_{t=1:T}$ is associated to all hidden layers.

FLDCRF-m model (see Fig. 3.5a) can be easily extended to accommodate multiple hidden layers for each label category $\{y_{i,t}\}_{t=1:T}$, $i = 1 : L$. However, for simplicity of model description, we assume only one hidden layer per label category in FLDCRF-m. This leads to a total L hidden layers in the model described below. We also assume first-order Markov connections among different hidden layers while applying FLDCRF in this thesis. Higher order connections are possible but avoided for simplicity. These extensions can be utilized in order to improve model performance at the cost of model complexity and more training data, however, we leave them to user discretion.

Let, $\mathbf{x} = \{x_1, x_2, \dots, x_T\}$ denote the sequence of observations. $\mathbf{y}_i = \{y_{i,1}, y_{i,2}, \dots, y_{i,T}\}$ are the observed labels along layer i , $i = 1 : L$. In the case of single-label prediction task by a FLDCRF-s (see Fig. 3.3b), all layers take the same labels during model training, i.e., $y_{i,t} = y_t, \forall i$. In the case of multi-agent interaction model (FLDCRF-i, see Fig. 3.5b), the features x_t are distributed across L layers, i.e., $x_t = \{x_{i,t}\}_{i=1:L}$, assuming L interacting agents in the environment.

Let, Υ_i be the alphabet of all possible class labels in layer i , $i = 1 : L$, i.e., $y_{i,t} \in \Upsilon_i$. $\mathbf{h}_i = \{h_{i,1}, h_{i,2}, \dots, h_{i,T}\}$ constitutes the i -th hidden layer. Each possible class label $\ell_i \in \Upsilon_i$ in layer i is associated with a set of hidden states $\mathcal{H}_{i,\ell_i}$. $\mathcal{H}_i$ is the set of all possible hidden states for layer i , given by, $\mathcal{H}_i = \bigcup_{\ell_i} \mathcal{H}_{i,\ell_i}$.

The joint conditional model $P(\{\mathbf{y}_i\}_{1:L} | \mathbf{x}, \theta)$ can be written as:

$$P(\{\mathbf{y}_i\}_{1:L} | \mathbf{x}, \theta) = \sum_{\{\mathbf{h}_i\}_{1:L}} P(\{\mathbf{y}_i\}_{1:L} | \{\mathbf{h}_i\}_{1:L}, \mathbf{x}, \theta) \cdot P(\{\mathbf{h}_i\}_{1:L} | \mathbf{x}, \theta). \quad (3.1)$$

In order to keep model computations during training and testing tractable, we introduce the layers of hidden variables $\{\mathbf{h}_i\}_{1:L} = \{h_{i,t}\}_{i=1:L, t=1:T}$ to the model with links (graphical constraints) as depicted in Fig. 3.7. This allows us to factorize $P(\{\mathbf{y}_i\}_{1:L} | \{\mathbf{h}_i\}_{1:L}, \mathbf{x}, \theta)$ according to:

$$P\left(\{\mathbf{y}_i\}_{1:L} \mid \{\mathbf{h}_i\}_{1:L}, \mathbf{x}, \theta\right) = \prod_{i=1}^L \prod_{t=1}^T P(y_{i,t} \mid h_{i,t}). \quad (3.2)$$

Figure 3.7: The general structure of FLDCRF with the essential graphical constraints. Hidden variables $h_{i,t}$, $i = 1 : L$, $t = 1 : T$ can assume any dependency structure (first-order Markov, second-order Markov, cotemporal links between layers etc.) within themselves in this general FLDCRF structure.

Replacing $P\left(\{\mathbf{y}_i\}_{1:L} \mid \{\mathbf{h}_i\}_{1:L}, \mathbf{x}, \theta\right)$ from equation (3.2), we can re-write equation (3.1) as:

$$\begin{aligned} P\left(\{\mathbf{y}_i\}_{1:L} \mid \mathbf{x}, \theta\right) &= \sum_{\{\mathbf{h}_i\}_{1:L} : \forall h_{i,t} \in \mathcal{H}_{i,y_{i,t}}} \left(\prod_{i=1}^L \prod_{t=1}^T P(y_{i,t} \mid h_{i,t}) \right) \cdot P\left(\{\mathbf{h}_i\}_{1:L} \mid \mathbf{x}, \theta\right) \\ &+ \sum_{\{\mathbf{h}_i\}_{1:L} : \exists h_{i,t} \notin \mathcal{H}_{i,y_{i,t}}} \left(\prod_{i=1}^L \prod_{t=1}^T P(y_{i,t} \mid h_{i,t}) \right) \cdot P\left(\{\mathbf{h}_i\}_{1:L} \mid \mathbf{x}, \theta\right). \end{aligned} \quad (3.3)$$

In order to further reduce model computations, we define the following model constraints, extending on [45]:

1. $\mathcal{H}_{i,\ell_i}$ are disjoint $\forall \ell_i \in \Upsilon_i$, $\forall i = 1 : L$.
2. $h_{i,t}$ can only assume values from the set of hidden states assigned to the label $y_{i,t}$, i.e., $h_{i,t} \in \mathcal{H}_{i,y_{i,t}}$, $\forall i = 1 : L$ and $\forall t = 1 : T$.

These constraints let us write the following:

$$P(y_{i,t} = \ell_i \mid h_{i,t}) = \begin{cases} 1, & h_{i,t} \in \mathcal{H}_{i,y_{i,t}=\ell_i} \\ 0, & h_{i,t} \notin \mathcal{H}_{i,y_{i,t}=\ell_i}. \end{cases} \quad (3.4)$$

Thus, the FLDCRF model in equation (3.3) can be reduced to:

$$P(\{\mathbf{y}_i\}_{1:L} \mid \mathbf{x}, \theta) = \sum_{\{\mathbf{h}_i\}_{1:L} : \forall h_{i,t} \in \mathcal{H}_{i,y_{i,t}}} P(\{\mathbf{h}_i\}_{1:L} \mid \mathbf{x}, \theta). \quad (3.5)$$

Equation (3.5) gives the general expression of a FLDCRF. This expression simplifies to a LDCRF [45] model for $L = 1$. If we assume only one distinct hidden state per class label in each layer i (equivalently, $|\mathcal{H}_{i,l_i}| = 1, \forall i, \forall l_i$; and $|\mathcal{H}_i| = |\Upsilon_i|, \forall i$), equation (3.5) simplifies to the joint conditional distribution $P(\{\mathbf{y}_i\}_{1:L} \mid \mathbf{x}, \theta)$ given by a DCRF [44] or a coupled CRF [67]. It is also straightforward to see that FLDCRF yields a LCCRF [42] when $L = 1$ and one distinct hidden state is associated to each class label. Thus, FLDCRF subsumes major sequential CRF variants viz., LDCRF, DCRF and LCCRF (illustrated in Fig. 3.8). We present a formal definition of FLDCRF in Section 3.4.1, based on above model construction process and the essential graphical and mathematical constraints.

Figure 3.8: Venn diagram illustrating the relationship between FLDCRF and major sequential CRF variants. The configuration underneath the model names describe (number of label categories, number of hidden layers, class alphabet and the sets of hidden states) in an FLDCRF-m model in order to derive it. For simplicity of description, we assume 1 hidden layer for each label category in the FLDCRF-m configuration, i.e., L hidden layers for L label categories.

We define $P(\{\mathbf{h}_i\}_{1:L} \mid \mathbf{x}, \theta)$ by the standard CRF formulation (see Chapter 2.3.1):

$$P(\{\mathbf{h}_i\}_{1:L} \mid \mathbf{x}, \theta) = \frac{1}{\mathbf{Z}(\mathbf{x}, \theta)} \exp \left(\sum_k \theta_k \cdot F_k(\{\mathbf{h}_i\}_{1:L}, \mathbf{x}) \right), \quad (3.6)$$

where index k ranges over all parameters $\theta = \{\theta_k\}$ and $\mathbf{Z}(\mathbf{x}, \theta)$ is the partition function given by:

$$\mathbf{Z}(\mathbf{x}, \theta) = \sum_{\{\mathbf{h}_i\}_{1:L}} \exp \left(\sum_k \theta_k \cdot F_k(\{\mathbf{h}_i\}_{1:L}, \mathbf{x}) \right). \quad (3.7)$$

Now we introduce first-order Markov assumptions (as described earlier) among the hidden variables $\{\mathbf{h}_i\}_{1:L}$. Therefore, the feature functions F_k 's are factorized (i.e., summed inside exponent) as:

$$F_k(\{\mathbf{h}_i\}_{1:L}, \mathbf{x}) = \sum_{t=1}^T f_k(\{h_{i,t-1}\}_{1:L}, \{h_{i,t}\}_{1:L}, \mathbf{x}, t). \quad (3.8)$$

We further factorize each $f_k(\{h_{i,t-1}\}_{1:L}, \{h_{i,t}\}_{1:L}, \mathbf{x}, t)$ at slice t , such that each hidden layer assumes its own (temporal) dynamics over the observations, while interacting with each other through cotemporal and first-order Markov (omitted in figures to avoid clutter) *influence* links. Thus, each factorized component of $f_k(\{h_{i,t-1}\}_{1:L}, \{h_{i,t}\}_{1:L}, \mathbf{x}, t)$ at slice t can be a *state* function $s_k(h_{i,t}, \mathbf{x}, t)$, a *transition* function $t_k(h_{i,t-1}, h_{i,t}, \mathbf{x}, t)$ or an *influence* function $i_k(h_{i,t-l}, h_{j,t-m}, \mathbf{x}, t)$, with $i, j \in \{1 : L\}$ and $(l, m) \in \{(0, 0), (0, 1), (1, 0)\}$. We define *state* and *transition* functions by the following combinations of linear and indicator functions:

$$\begin{aligned} s_k(h_{i,t}, \mathbf{x}, t) &= \mathbb{1}_{\{(h_{i,t})=k\}} \cdot x_t, \\ t_k(h_{i,t-1}, h_{i,t}, \mathbf{x}, t) &= \mathbb{1}_{\{(h_{i,t}, h_{i,t-1})=k\}}, \end{aligned} \quad (3.9)$$

assuming one-hot representations for discrete x_t components. We define *influence* functions $i_k(h_{i,t-l}, h_{j,t-m}, \mathbf{x}, t)$, with $i, j \in \{1 : L\}$ and $(l, m) \in \{(0, 0), (0, 1), (1, 0)\}$ as:

$$i_k(h_{i,t-l}, h_{j,t-m}, \mathbf{x}, t) = \mathbb{1}_{\{(h_{i,t-l}, h_{j,t-m})=k\}}. \quad (3.10)$$

For the interaction model depicted in Fig. 3.5b, the mathematical expressions are identical, only with a minor change in the *state* function:

$$s_k(h_{i,t}, \mathbf{x}_i, t) = \mathbb{1}_{\{(h_{i,t})=k\}} \cdot x_{i,t}, \quad (3.11)$$

where $\mathbf{x}_i = \{x_{i,t}\}_{t=1:T}$.

3.4.1 Definition: FLDCRF

Let, $\mathbf{x} = \{x_1, x_2, \dots, x_T\}$ denote the sequence of observations and $\mathbf{y}_i = \{y_{i,1}, y_{i,2}, \dots, y_{i,T}\}$ denote the observed labels along layer i , $i = 1 : L$.

Let, Υ_i be the alphabet of all possible class labels in layer i , $i = 1 : L$, i.e., $y_{i,t} \in \Upsilon_i$. Let, $\mathbf{h}_i = \{h_{i,1}, h_{i,2}, \dots, h_{i,T}\}$ denote the i -th hidden layer, $i = 1 : L$. Each possible class label $\ell_i \in \Upsilon_i$ in layer i is associated with a set of hidden states $\mathcal{H}_{i,\ell_i}$. $\mathcal{H}_i$ is the set of all possible hidden states for layer i , given by, $\mathcal{H}_i = \bigcup_{\ell_i} \mathcal{H}_{i,\ell_i}$.

Then, the joint conditional model $P(\{\mathbf{y}_i\}_{1:L} \mid \mathbf{x}, \theta)$ written as:

$$P(\{\mathbf{y}_i\}_{1:L} \mid \mathbf{x}, \theta) = \sum_{\{\mathbf{h}_i\}_{1:L}} P(\{\mathbf{y}_i\}_{1:L}, \{\mathbf{h}_i\}_{1:L} \mid \mathbf{x}, \theta), \quad (3.12)$$

is a FLDCRF iff,

1. The variables $\mathbf{x}$, $\{\mathbf{y}_i\}_{1:L}$, and $\{\mathbf{h}_i\}_{1:L}$ obey the essential graphical constraints depicted in Fig. 3.7,
2. $\mathcal{H}_{i,\ell_i}$ are disjoint $\forall \ell_i \in \Upsilon_i, \forall i = 1 : L$, and
3. $h_{i,t}$ only assumes values from the set of hidden states assigned to the label $y_{i,t}$, i.e., $h_{i,t} \in \mathcal{H}_{i,y_{i,t}}, \forall i = 1 : L$ and $\forall t = 1 : T$.

3.4.2 Training

We estimate the model parameters by maximizing the conditional log-likelihood of the training data:

$$\mathbf{L}(\theta) = \sum_{n=1}^N \log P(\{\mathbf{y}_i\}_{1:L}^{(n)} \mid \mathbf{x}^{(n)}, \theta) - \frac{\|\theta\|^2}{2\sigma^2}, \quad (3.13)$$

where N is the total number of available labeled sequences. A L2 regularizer (second term in equation (3.13)) was included in order to reduce overfitting during our experiments.

The factors $P(\{\mathbf{y}_i\}_{1:L}^{(n)} \mid \mathbf{x}^{(n)}, \theta)$, $n = 1 : N$ are obtained from equations (3.5)-(3.10). The numerator $\mathbf{N}(\mathbf{x}, \theta) = \sum_{\{\mathbf{h}_i\}_{1:L} : \forall h_{i,t} \in \mathcal{H}_{i,y_{i,t}}} \exp\left(\sum_k \theta_k \cdot F_k(\{\mathbf{h}_i\}_{1:L}, \mathbf{x})\right)$ and the denominator $\mathbf{Z}(\mathbf{x}, \theta)$ of equation (3.5), given by (3.6), (3.7) and (3.8), are in the classic sum-product form of dynamic programming and are efficiently computed by the forward algorithm ([46], see Chapter 2.6). We apply the default BFGS optimizer in Stan modeling language [31] to obtain the estimates $\hat{\theta} = \{\hat{\theta}_k\}$.

3.4.3 Inference

Multiple label sequences $\mathbf{y}_i$, $i = 1 : L$, can be inferred from the same graph structure by marginalizing over other labels:

$$\hat{\mathbf{y}}_i = \operatorname{argmax}_{\mathbf{y}_i} \sum_{\{\{\mathbf{y}_i\}_{1:L} - \mathbf{y}_i\}} P(\{\mathbf{y}_i\}_{1:L} \mid \mathbf{x}, \hat{\theta}), \quad (3.14)$$

where $\mathbf{x} = \{x_t\}_{t=1:T}$ is the observed input sequence of length T . $P(\{y_i\}_{1:L} | \mathbf{x}, \hat{\theta})$ can be obtained from (3.5) and estimated parameters $\hat{\theta}$. For example, at each instant t , the marginals $P(\{h_{i,t}\}_{1:L} | \mathbf{x}, \hat{\theta})$ are computed and summed according to the disjoint sets of hidden states to obtain joint estimates of desired labels $\hat{y}_{i,t}$, $t = 1, 2, \dots$, $\forall i = 1 : L$, as follows:

$$P(\{y_{i,t}\}_{1:L} | \mathbf{x}) = \sum_{\{h_{i,t}\}_{1:L} : h_{i,t} \in \mathcal{H}_{i,y_{i,t}}} P(\{h_{i,t}\}_{1:L} | \mathbf{x}, \hat{\theta}). \quad (3.15)$$

After marginalizing according to (3.14), the label $\hat{y}_{i,t}$ corresponding to the maximum probability is inferred. Since we consider online inference (i.e., input sequence observed upto current instant t) throughout the thesis, the inference problem in equation (3.15) gets reduced to $P(\{y_{i,t}\}_{1:L} | x_{1:t})$. We compute the necessary probabilities $P(\{h_{i,t}\}_{1:L} | x_{1:t}, \hat{\theta})$ by the forward algorithm ([46], see Chapter 2.6). Forward-backward algorithm [46] and Viterbi algorithm [77] can also be applied for problems where online inference is not necessary, i.e., processing delays in prediction are tolerable.

3.5 Datasets

We apply our models on two datasets: a) UCI gesture phase segmentation dataset [76], and b) UCI opportunity dataset [73]. We describe the datasets below.

3.5.1 UCI Gesture Phase Dataset

The UCI gesture phase data consists of features extracted from 7 videos with people gesticulating while telling stories. The data is captured by a Microsoft Xbox Kinect. Each time point within the 7 sequences is described by 18 positional features, 32 dynamic features (velocity, acceleration etc.) and is labeled with one of the five gesture classes: rest, preparation, hold, stroke and retraction. The number of instances in each sequence is tabulated in Table 3.1. ‘A’, ‘B’, and ‘C’ denote different participants and ‘1’, ‘2’, ‘3’ describe different stories. Although the data is multiclass, the most popular studies on the dataset [76, 78] have only considered binary classification, i.e., segmentation of gesture positions (i.e., positives - preparation, hold, stroke, retraction) from the rest (negative) position. We perform two binary time-series continuous gesture segmentation experiments on this dataset:

1. We follow the experiment 2 in [78]. The goal of the experiment is to train on the sequence A1 and test on sequence A2. We utilize the ‘Data vector 2’ in the

paper [78] as our input features x_t , which consists of 3 dimensional positions of the left hand, right hand, left wrist and right wrist together, yielding a 12 dimensional input feature set. During experiment, we selected our models (hyperparameters, see Section 3.6) on a validation set comprising of the last 30% frames of sequence A1, while training on the first 70%. During testing, we re-trained the selected models on the entire A1 sequence.

2. In order to increase training and test data and for better generalization of model performances on the dataset, we also perform a nested cross validation, with 7 outer loops (one for testing each sequence) and 6 inner loops (leave-one-out) per outer loop. We present results of both our experiments on the UCI gesture phase data in Section 3.7.2.

Table 3.1: Number of instances in each sequence in UCI gesture phase dataset [76].

Sequence	A1	A2	A3	B1	B3	C1	C3	Total
Num frames	1747	1264	1834	1073	1423	1111	1448	9900

3.5.2 UCI Opportunity Dataset

The UCI opportunity dataset [73] is a public activity recognition dataset. The data contains 20 ADL (activity of daily living) sequences from 4 different participants (S1, S2, S3 and S4). Each participant has 5 ADL sequences. Each instance in the data has 3 activity label categories:

- locomotion, which has four classes, viz., ‘lie’, ‘sit’, ‘stand’ and ‘walk’;
- a high-level activity (HL), which has five classes, viz., ‘relaxing’, ‘early morning’, ‘coffee time’, ‘sandwich time’ and ‘clean up’; and
- a mid-level activity (ML), which has several classes.

Figure 3.9: Illustrating each ADL sequence in the UCI opportunity dataset [73].

We do not utilize the mid-level activity labels in this thesis. The 5 high-level activities are performed in sequence within each complete ADL run (see Fig. 3.9). We consider the 5 sequences (ADL runs) from participant S2 in our experiment. From each ADL sequence, we utilize the data upto two high-level activity classes (i.e., ‘Relaxing’ and ‘Early morning’), illustrated in Fig. 3.9. The ‘Relaxing’ portions of the sequence primarily consists of the ‘lie’ and ‘sit’ locomotion classes, while the ‘early morning’ activity class comes mostly together with ‘stand’ and ‘walk’ locomotion classes, making the two label categories depending on each other. Each time-point in the data also contains 242 sensor outputs (113 body-worn sensors, 32 shoe sensors and 97 other object and ambient sensors). We utilize the body-worn and shoe sensor outputs as our input features, yielding a 145 dimensional input feature set. For our convenience, we further divided the data into 28 sub-sequences, each fully labeled by the two label categories. Additionally, there are instances within the data with missing values for some sensor outputs. We replaced such values with the previous instant, if available, otherwise by ‘0’s. Since the FLDCRF models encode the input features (x_t) by exponential functions over $(\theta_k \cdot x_t)$, replacing values by ‘0’s means a multiplication by 1, thus not affecting the model likelihood. We preprocessed each input dimension of x_t between 0 to 1 for faster learning of our models.

We perform a nested cross-validation with 5 outer loops (with test sequences coming from each ADL sequence) and 4 inner loops (leave-one-ADL sequence out) per outer loop. Tabulated data (features and labels) for our multi-label sequence tagging experiment on UCI opportunity data is available here². Table 3.2 provides the details (length, ADL run etc.) of the 28 sub-sequences considered in this experiment.

Important dataset characteristics of the two datasets are listed in Table 3.3.

3.6 Experimental Setup

3.6.1 Metrics for evaluation

We assess the different models by the F1-score. For the gesture phase segmentation problem on UCI gesture phase dataset, we consider rest position as ‘negative’ and other gestures as ‘positive’. The F1 measure (F) is computed as:

$$F = \frac{2 \cdot P \cdot R}{P + R}, \quad (3.16)$$

where P and R represent precision and recall, defined as $P = \frac{TP}{TP+FP}$ and $R = \frac{TP}{TP+FN}$ respectively. TP , FP and FN are the predicted true positives, false positives and

²<https://github.com/satyajitneogiju/FLDCRF-for-sequence-labeling>.

Table 3.2: Number of instances in each sequence in the nested CV experiment on UCI opportunity dataset [73].

Sequence Nested outer set	Sequence									
	1	2	3	4	5	6	7	8	9	Total
1	1000	1000	800	943	724	847	1352	1276	298	8240
2	721	863	800	1428	216					4028
3	583	843	800	746	655					3627
4	477	253	1008	954	583					3275
5	711	1338	626	493						3168
Total										22338

false negatives by the model.

For the multi-class action recognition problem on UCI opportunity data, we assess the models by the micro F1-score (F_{micro}), which also is defined by (3.16), where precision and recall are given by:

$$P_{micro} = \frac{\sum_{c=1}^{N_c} TP_c}{\sum_{c=1}^{N_c} TP_c + \sum_{c=1}^{N_c} FP_c}, \quad R_{micro} = \frac{\sum_{c=1}^{N_c} TP_c}{\sum_{c=1}^{N_c} TP_c + \sum_{c=1}^{N_c} FN_c}, \quad (3.17)$$

where N_c is the number of classes. We apply the micro F1-score (F_{micro}) for evaluating the ‘locomotion’ (4 classes) and overall (6 classes) performance. As both class labels (‘relaxing’ and ‘early morning’) of the high-level activity (HL) are equally important, we slightly modify the precision (P_{HL}) and recall (R_{HL}) as follows: $P_{HL} = \frac{TP_{early} + TP_{relax}}{TP_{early} + TP_{relax} + FP_{early}}$ and $R_{HL} = \frac{TP_{early} + TP_{relax}}{TP_{early} + TP_{relax} + FN_{early}}$, while assessing individually on the HL activity.

During validation, we select the multi-label sequential models (FLDCRF-m and LSTM-m) based on their combined performance (on all 6 classes) in the inner loops. The single-label models are separately selected on each label category.

3.6.2 Benchmarking

For the single label sequence tagging tasks (‘experiment 2’ [78] and nested CV) on the UCI gesture phase data, we compare CRF, LDCRF, FLDCRF-s, LSTM, and LSTM-CRF. For the multi-label sequence tagging task, we compare CRF, LDCRF, FLDCRF-s, LSTM, LSTM-CRF, FCRF, CCRF, FLDCRF-m, and LSTM-m.

Table 3.3: Information on the UCI gesture phase [76] and the UCI opportunity dataset [73].

Attribute \ Data	UCI gesture phase	UCI opportunity
Type	Single label	Multi label
No. sequences	7	5
No. sub-sequences	7	28
Experiments	Experiment 2 [78], Nested LOOCV	Nested LOOCV
No. labels	1 (Gesture)	2 (Locomotion & High-level activity)
No. classes/label	2	4 & 2
Tested models	CRF, LDCRF, FLDCRF-s, LSTM, LSTM-CRF	CRF, LDCRF, FLDCRF-s, LSTM, LSTM-CRF, FCRF, CCRF, FLDCRF-m, LSTM-m
Input dimension ($ x_t $)	12	145
Metrics	F1-score	Micro F1-score

A FLDCRF-s model has two hyperparameters: number of hidden layers N_h and number of hidden states per class label $\{N_{si}\}$ along layers $i = 1 : L$. For simplicity, we apply the same number of hidden states along all layers, i.e., $N_{si} = N_s, \forall i = 1 : L$. We denote such model by FLDCRF-s($\langle N_h \rangle / \langle N_s \rangle$). For example, if we consider 2 hidden layers and 3 hidden states (per class label) across each layer, then the FLDCRF-s model will be denoted by FLDCRF-s(2/3). In Table 3.4, we provide the list of tested hyperparameter settings of FLDCRF-s across the 3 different experiments.

A FLDCRF-m (multi-label) model has three hyper-parameters: number of different label categories N_l , number of hidden layers per label category $\{N_{hl}\}_{l=1:N_l}$, and number of hidden states per class label $\{N_{si}\}$ along layers $i = 1 : L$, where $L = \sum_{l=1}^{N_l} \{N_{hl}\}$. For simplicity, we only keep one hidden layer ($N_{hl} = 1, \forall l = 1 : N_l$) for each label category, thus making total number of hidden layers equal to the number of different label categories, i.e., $L = N_l$. We denote such a model by FLDCRF-m $\{N_{si}\}_{i=1:L}$. For example, if there are 3 label categories and we associate 1, 2 and 4 hidden states (per class label) respectively to each label category, then the 3-layered

FLDCRF-m model will be represented as FLDCRF-m $\{1, 2, 4\}$. In our multi-label experiment on the opportunity data, there are two different label categories (locomotion and high-level), and the FLDCRF-m models are denoted as FLDCRF-m $\{N_{s1}, n_{s2}\}$. In Table 3.4, we list the considered hyperparameter settings of FLDCRF-m. We considered two different FLDCRF-m models (with and without the first-order Markov influence among hidden layers, see Fig. 3.10) during our multi-label experiment on the UCI opportunity data.

Figure 3.10: Two different FLDCRF-m models considered in the multi-label experiment on the UCI opportunity data.

An LSTM has three hyperparameters: number of hidden units (N_{hls}), number of epochs to train (N_{els}), and training minibatch size (N_{mls}). To the best of our knowledge, there is no generally accepted rule to select the optimal N_{hls} , therefore, they need to be tuned by trial and error. As suggested in [79, 80, 81], we vary the N_{hls} according to,

$$N_{hls} = \frac{N_{sa}}{\alpha \cdot (N_i + N_o)}, \quad (3.18)$$

where N_{sa} is the number of samples in the training dataset (number of instances; see Tables 3.1 and 3.2), N_i is the number of input neurons ($|x_t|$ in our case; see Table 3.3) and N_o is the number of output neurons from the LSTM layer. We vary the parameter α between 2 and 10 to select the N_{hls} 's. A few other popular suggestions [79, 80] include:

- Keep N_{hls} between N_i and N_o ,

- Set N_{hls} as the arithmetic mean of N_i and N_o ,
- Set N_{hls} as the geometric mean of N_i and N_o etc.

We try to follow these recommendations as closely as possible while selecting N_{hls} values to be tested. In Table 3.4, we list the considered LSTM N_{hls} values in each of our experiments. During validation, we run each model setting N_{hls} for 500 epochs, saving model performances on validation sets at every 100 epochs. We feed all training sequences to the LSTM models at each epoch at the rate of 1 sequence per minibatch.

Table 3.4: Hyperparameter settings for FLDCRF-s, FLDCRF-m, LSTM and LSTM-m for experiments in this chapter. We consider same hyperparameters for LSTM and LSTM-CRF.

Experiment	Models			
	FLDCRF-s (N_h/N_s)	FLDCRF-m ($\{N_{s1}, n_{s2}\}$)	LSTM (N_{hls})	LSTM-m (N_{hls})
Experiment 2 [78]	1/1, 1/2, 1/3, 1/4, 1/5, 1/6, 1/7, 2/1, 2/2, 2/3, 2/4, 3/1, 3/2, 3/3	NA	2, 5, 7, 10, 14, 20, 50, 100, 200	NA
Nested CV on Gesture phase	1/1, 1/2, 1/3, 1/4, 1/5, 1/6, 2/1, 2/2, 2/3, 2/4, 2/5, 3/1, 3/2, 3/3	NA	2, 5, 7, 10, 14, 20, 50, 100, 200	NA
Nested CV on Opportunity (Locomotion)	1/1, 1/2, 1/3, 1/4, 2/1, 2/2, 2/3	{1,1}, {1,2}, {1,3}, {1,4}, {1,5}, {1,6}, {2,1}, {2,2}, {2,3}, {2,4}, {2,5}, {2,6}, {3,1}, {3,2}, {3,3}	5, 10, 25, 50, 75, 150, 300, 500	5, 10, 25, 50, 75, 150, 300, 500
Nested CV on Opportunity (HL)	1/1, 1/2, 1/3, 1/4, 1/5, 1/6, 2/1, 2/2, 2/3, 2/4		2, 5, 10, 25, 50, 75, 150	

3.6.3 Model Platforms and System Specifications

We trained and tested FLDCRF type models (FLDCRF, LDCRF, CRF, CCRF etc.) by the PyStan interface of Stan modeling language [31]. Stan's in-built BFGS and L-BFGS optimizers allow us to conveniently train models by defining the model likelihoods. We apply the default BFGS optimizer [31] to train all FLDCRF models and keep all default settings for the optimizer.

We implemented the LSTM models [1] defined in Keras, a deep learning library for Python running Tensorflow in the backend. LSTM parameters were trained by an Adam optimizer with the default learning rate of 0.001 and other default parameters. To reduce overfitting, we added a dropout layer (with regularization 0.2) between the LSTM and softmax (or CRF for LSTM-CRF model) layers for all models with ≥ 5 hidden units. We also added a dropout (with regularization 0.2) to the LSTM layer for all models with ≥ 5 hidden units. Our Pystan FLDCRF and Keras LSTM implementation codes are available from - <https://github.com/satyajitneogiju/FLDCRF-for-sequence-labeling>.

We perform the metric computations and plot figures in MATLAB 2015b [36]. The LSTM models are trained (and tested) on an Nvidia Tesla K80 GPU, while the FLDCRF models are trained (and tested) on an Intel(R) Xeon(R) CPU E5-2630 v3 @2.40GHz CPU.

3.7 Results

A good model should not only perform well on the validation data, it must be consistent across validation and test data. At the same time, the model selection process should be lucid and easy. It is also desirable to have fast model training and inference mechanisms, preferably without additional resources, e.g., GPU. Therefore, we not only compare different models (FLDCRF, LSTM, LSTM-CRF etc.) on the test data, but also examine several other modeling aspects, viz., ease of model selection, consistency across validation and test performance, and computation times.

3.7.1 Performance Attributes

We present the following 4 types of model performance measures for each experiment:

- *Test performance:* Under this measure, different models are compared on the test data.

- *Ease of model selection:* Under this performance attribute, we discuss the model hyperparameter selection process of FLDCRF and LSTM families on the validation data. In order for this model selection process to be fast, easy and effective, a model must have the following characteristics:
 1. Few hyperparameters: There should be only a few hyperparameters that strongly influence the model performance. FLDCRF models only depend on the N_h/N_s settings, and do not need to be tuned for the number of training epochs. LSTM models, on the other hand, need to be optimized for the number of training epochs alongside the number of hidden units N_{hls} and training minibatch size N_{mls} .
 2. Rule to choose hyperparameters: It is best to have definite rules to choose the model hyperparameters, rather having to select them via trial and error.
 3. Observable pattern in validation performance: In order to easily select the model hyperparameters, the model must exhibit discernible patterns (increasing/ decreasing) in validation performance against the choices of the hyperparameters. Model performance across different hyperparameter settings should not vary at random.
 4. Small variance: In order to avoid tedious hyperparameter selection and continuous monitoring of model performance on validation data, model performance across adjacent hyperparameter settings should not vary widely. Wide and random variation across hyperparameters (on validation data) without any pattern also require widespread selection of hyperparameters.
 5. Reasonable worst-case performance: The model must guarantee good performance with minimal effort to optimize hyperparameters. To accomplish this, worst-case performances reported by the model on validation data should not differ largely from the best/average performances, as well as percentage of such poor outcomes should be low (if any).
- *Consistency:* We examine consistency of the selected models on test data under this measure. Consistency in model performance across validation and test data reflects stability of a model for practical deployment.
- *Computation times:* In this result category, we present computation times required by the models for training and inference, for different choices of the hyperparameters.

3.7.2 UCI Gesture Phase Dataset

As mentioned earlier, we perform two experiments (see Section 3.5.1) on the UCI Gesture Phase Dataset for segmenting gestures from rest positions. In the first experiment, we follow the experiment 2 by [78]. In the second experiment, we perform a 7 (outer) fold nested cross-validation.

Experiment 1: [78]

- *Test performance:*

Table 3.5 compares different models on the test sequence A2 of the UCI gesture phase dataset. FLDCRF-s considerably outperforms LSTM and LSTM-CRF models on the test set. FLDCRF-s also outshines a multi-layered perceptron (MLP) model [78] reported on the test data. However, FLDCRF-s does not improve LD-CRF performance in this experiment.

Considering variable performance on re-training the LSTM models, we also report performance of the optimized LSTM-CRF model that is not re-trained on the combined training and validation data. FLDCRF-s outperforms this LSTM-CRF model (F1-score 85%) as well on the test data.

Table 3.5: F1-scores of the models for the experiment proposed by [78] on UCI gesture phase data. The best LSTM-CRF model on the validation set achieved a F1-score 84.88 without retraining on the training+validation set.

Model	F1-score (%)
CRF	81.35
LD-CRF	88.60
FLDCRF-s	88.60
LSTM	83.42
LSTM-CRF	84.28 (84.88)
Multi-layered Perceptron [78]	82.34

- *Model selection:*

FLDCRF-s models only need to be optimized for number of hidden layers N_h and number of hidden states N_s . Each LSTM model (LSTM and LSTM-CRF,

with a given number of hidden units N_{hls}), on the other hand, need to be tuned for the optimum number of training epochs, thus requiring careful monitoring on validation performance in order to avoid overfitting. We analyze the selection process of the best performing models in each family, viz., FLDCRF-s and LSTM-CRF in the following.

Table 3.6 presents FLDCRF-s performance on validation data. The models continue to perform better on increasing N_s upto 6 with $N_h = 1$, but decline on increasing N_h , giving clear indications to stop testing more hyperparameters. On the other hand, LSTM-CRF model performances vary rapidly across the training epochs (see Table 3.7) for most of the N_{hls} settings. While in some cases (e.g., $N_{hls} = 10, 14$ etc.) model performance deteriorates rapidly after attaining the maximum, there are cases (e.g., $N_{hls} = 100, 200$ etc.) where the validation F1-score varies randomly across training epochs, without any discernible pattern against the hyperparameter. Such random variation demands very careful monitoring over the validation outcomes. Even when optimized for the training epochs for each N_{hls} setting (see row 7 of Table 3.7), F1-scores do not reveal any pattern over the N_{hls} 's. Such a behaviour additionally brings in the problem of considering widespread values of N_{hls} .

Table 3.6: FLDCRF-s validation performance on experiment 2 [78] on UCI gesture phase data.

FLDCRF-s	1/1	1/2	1/3	1/4	1/5	1/6	1/7	Best	Worst	Std
F1	80.07	80.79	80.87	81.31	81.6	82.96	81.92	82.96	76.47	2.11
FLDCRF-s	2/1	2/2	2/3	2/4	3/1	3/2	3/3			
F1	77.57	77.84	78.97	78.03	76.84	76.60	76.47			

The F1-scores of FLDCRF-s models do not vary rapidly (standard deviation 2.11) across the choices of the hyperparameters, and rather indicate a worst-case F1-score of 76.47%. On the other hand, a high percentage (18%) of the LSTM-CRF models generate poor F1-scores ($<75\%$) with a notably low worst case performance (62%). In addition, LSTM-CRF models display a considerably higher standard deviation (5.11) among validated models.

The selection of hyperparameters for LSTM models is challenging, since the performance tends to vary substantially across hyperparameter settings without patterns and the worst-case performance is unacceptably poor. We observe similar behavior for LSTM models in all our experiments.

Table 3.7: LSTM-CRF validation performance on experiment 2 [78] on UCI gesture phase data. N_{els} denotes the number of training epochs and N_{hls} represents the number of hidden units in the model.

$N_{els} \backslash N_{hls}$	2	5	7	10	14	20	50	100	200	Best	Worst	Std
100	68.9	78.7	78.8	81.3	80	83.9	73.4	78.8	72.1	87.17	62.45 (18%<75)	5.59
200	75	81.1	82.5	71.9	75.6	76.5	81.4	68.2	86.4			
300	79.9	80.1	83.6	67.4	87.2	80.3	82.8	75.9	77.7			
400	81.8	78.6	83.4	63.3	80.2	79.1	77.3	75.3	84.7			
500	83.4	79.4	84.2	62.5	76.5	81.2	77.8	75.1	76.9			
Best	83.4	81.1	84.2	81.3	87.2	83.9	82.8	78.8	86.4			
Std	5.9	1.01	2.17	7.69	4.54	2.71	3.7	3.9	5.88			

- *Consistency:*

Although FLDCRF-s offers easier model selection than LSTM-CRF, LSTM-CRF outperforms FLDCRF-s on the best validation performance, yielding an F1-score 87% compared to 83% by FLDCRF-s. However, this optimized LSTM-CRF model, when re-trained with same hyperparameters on training+validation data, fails to outperform FLDCRF-s on the test data (see Table 3.5), achieving only 84.3% compared to 88.6% by FLDCRF-s. Furthermore, considering variable performance for LSTM-CRF on re-training the model, we apply the best (pre-trained) LSTM-CRF model (with 87% on validation) on the test data. However, this model (F1-score 84.9%) also fails to outperform FLDCRF-s on the test data (see Table 3.5).

In addition to a tedious model selection process, LSTM-CRF displays inconsistent performance across validation and test data. This inconsistency is reflected throughout the experiments in the paper.

- *Computation times:*

We compare the training and inference times required by the FLDCRF and LSTM models in Fig. 3.11. While few FLDCRF-s models and LSTM required similar computation times for inference, the computation time required to train the FLDCRF-s models is notably lower than the LSTM models. Furthermore, the FLDCRF computation times can be significantly reduced by GPU implementation and careful optimization, allowing more hidden layers (N_h) and states (N_s)

for large datasets, if necessary. We consider the GPU implementation and application of FLDCRF on big data as our future work.

Figure 3.11: Training and inference times of different FLDCRF and LSTM models on UCI gesture phase data. Training time shown for each FLDCRF-s is until model convergence on training data, with the default criteria in Stan. Training time shown for LSTM is for 500 training epochs.

Experiment 2: Nested CV

Since the training and test datasets in Experiment 1 were small, we consider the entire UCI gesture phase dataset in this experiment. We perform nested cross-validation. Since there are 7 large sequences in the dataset (see Section 3.5.1), we divide the data in 7 outer folds, one for testing each sequence. In each case, we select the models by 6-fold (leave-one-out) cross-validation on the remaining sequences.

- *Test performance:*

Table 3.8 compares different models on the test sets. FLDCRF-s boosts LDCRF performance on 4 sets (sets 1, 2, 3, 7), with notable improvements ($\sim 3\%$ and $\sim 2\%$) on sets 1 and 3. As a result, FLDCRF-s improves the overall LDCRF performance to outshine the overall LSTM (and LSTM-CRF) performance on the test sets. FLDCRF-s individually outperforms LSTM and LSTM-CRF models on 4 (out of 7) sets each.

- *Model selection:*

Tables 3.9 and 3.10 present FLDCRF-s and LSTM cross-validation F1 outcomes on sets 4 and 7 respectively. Similar to experiment 1, LSTM models display rapid variation in performance across training epochs, demanding careful tracking of validation performance. Few such instances are highlighted in Table 3.10. FLDCRF-s

Table 3.8: F1-scores on the nested CV experiment on UCI gesture phase data.

Model	F1-score							
	1	2	3	4	5	6	7	Average
LDCRF	85.60	76.92	87.81	92.60	92.76	91.83	76.10	86.23
FLDCRF-s	88.40	76.95	89.69	92.60	92.76	91.38	76.17	86.85
LSTM	90.95	81.61	82.59	94.41	92.51	89.12	73.50	86.39
LSTM-CRF	91.99	81.51	81.60	93.40	92.76	86.87	74.06	86.03

does not require to tune the training epochs, and variation among adjacent hyperparameters (see Table 3.9) is neither rapid nor random, with observable decay in performance due to overfitting beyond certain settings of the hyperparameters.

Table 3.9: FLDCRF-s validation performance on nested cross-validation sets 4 and 7 of the nested CV experiment on UCI gesture phase data.

Set 4									
FLDCRF	1/1	1/2	1/3	1/4	1/5	1/6	Best	Worst	Std
F1	86.38	85.58	86.73	86.5	86.33	85.96	86.73	85.01	0.50
FLDCRF	2/1	2/2	2/3	2/4	3/1	3/2			
F1	86.44	86.44	86.5	85.01	86.6	85.89			
Set 7									
FLDCRF	1/1	1/2	1/3	1/4	1/5	1/6	Best	Worst	Std
F1	88.3	89.05	89.19	88.7	88.52	88.96	89.23	88.3	0.28
FLDCRF	2/1	2/2	2/3	2/4	3/1	3/2			
F1	89.05	88.94	89.23	89.11	88.94	89.09			

Table 3.11 highlights the overall FLDCRF-s and LSTM cross-validation performance on the inner loops (for selecting models). We report the best, worst-case performances and standard deviation among models on validation sets. We also report percentage of poor performing models, i.e., validated models (hyperparameters) below a given threshold (F1-score 85%). FLDCRF-s produces the better worst-case performance with smaller standard deviation among models than LSTM in most of the sets.

Wide and random variance in F1-scores from LSTM models demand careful mon-

Table 3.10: LSTM validation performance on nested validation sets 4 and 7 of the nested CV experiment on UCI gesture phase data. N_{els} denotes number of epochs trained and N_{hls} represents the number of hidden units in the model.

Set 4												
$N_{els} \backslash N_{hls}$	2	5	7	10	14	20	50	100	200	Best	Worst	Std
100	86.3	86.7	86.3	87.2	85.6	83.4	84.6	85.3	84.6	87.96	81.64	1.43
200	85.3	87.3	86.3	84.6	85.6	86.6	82.5	86.0	83.9			
300	85.0	87.8	85.5	87.9	85.3	87.5	85.5	86.6	85.5			
400	84.8	85.7	86.2	86.7	85.9	87.9	83.5	86.3	85.7			
500	85.5	86.0	84.7	86.1	85.3	83.1	83.5	81.6	86.4			
Std	0.59	0.71	0.69	1.23	0.28	2.31	1.14	2.05	0.99			
Set 7												
$N_{els} \backslash N_{hls}$	2	5	7	10	14	20	50	100	200	Best	Worst	Std
100	86.6	86.9	86.2	86.4	86.6	86.8	87.7	90.1	88.6	90.72	80.31	1.65
200	86.5	88.5	86.6	85.7	86.7	89.3	86.7	88.2	87.7			
300	88.3	89.0	87.2	86.8	85.9	90.1	86.2	88.3	87.8			
400	87.3	87.9	87.5	86.5	86.9	90.7	86.9	88.1	87.4			
500	85.7	88.7	88.2	87.2	87.6	80.3	87.0	87.1	88.2			
Std	0.99	0.83	0.78	0.56	0.59	4.27	0.55	1.08	0.47			

itoring of performance (over N_{els}), as well as require widespread selection of N_{hls} settings, making is tedious to optimize the LSTM models. A high percentage of poor validation F1-scores and their random presence make this process even harder. 17.1% of all considered LSTM models across 7 cross-validation sets report below F1-score 85, as compared to only 5% for FLDCRF-s models. In set 7, although 2.2% of LSTM models report F1 below 85, a staggering 69% of the models perform below F1-score 88, while FLDCRF-s has a worst case F1 of 88.3.

- *Consistency:*

LSTM models outperform FLDCRF-s models during cross-validation (see F1-score (best) in Table 3.11) on most (6 out of 7) of the sets. However, such optimized models fail to consistently outperform FLDCRF-s on the test sets (see Table 3.8),

Table 3.11: Summary of cross-validation performance by FLDCRF-s and LSTM on the inner loops of nested CV experiment on the UCI gesture phase data.

Model	F1-score (best)						
	1	2	3	4	5	6	7
FLDCRF-s	87.44	86.55	88.88	86.73	88.59	86.32	89.23
LSTM	88.24	87.32	89.85	87.96	86.71	87.27	90.72
Model	F1-score (worst)						
	1	2	3	4	5	6	7
FLDCRF-s	86.02	84.38	84.89	85.01	86.91	85.03	88.3
LSTM	84.24	82.17	85.17	81.64	82.24	78.58	80.31
Model	F1-score (std)						
	1	2	3	4	5	6	7
FLDCRF-s	0.38	0.83	1.29	0.55	0.47	0.50	0.27
LSTM	0.90	0.70	1.07	1.43	0.94	1.36	1.65
Model	# F1-score < 85 (% of all validated models)						
	1	2	3	4	5	6	7
FLDCRF-s	0	28.5	7.14	0	0	0	0
LSTM	8.9	20	0	26.7	37.8	24.4	2.2 ³

and manage to perform better only on 3 (out of 7) sets. The average performance reported by LSTM on the test data is also about 0.5% less than FLDCRF-s.

- *Computation times:*

Figure 3.12 compares considered FLDCRF-s and LSTM training times. LSTM training times are shown for 500 training epochs, while the FLDCRF-s training times are until convergence (by the default criteria for BFGS in Stan). Displayed results are quite similar to experiment 1, only magnified by the amount of training data. As mentioned earlier, FLDCRF training times can be further reduced by GPU implementation.

³68.9% of the LSTM models below F1-score 88, while FLDCRF-s reports worst-case F1 of 88.3.

Figure 3.12: Average training time required by different FLDCRF settings per outer fold of the nested CV experiment on the UCI gesture phase data.

3.7.3 UCI Opportunity Dataset

We perform the multi-label sequence tagging experiment on the UCI Opportunity Dataset. We compare several single-label (CRF, LDCRF, FLDCRF-s, LSTM, LSTM-CRF) and multi-label (FCRF, CCRF, FLDCRF-m1, FLDCRF-m2, LSTM-m) state-of-the-art models in the nested CV experiment.

Experiment 1: Nested CV

This experiment has 5 outer loops, one for testing on each ADL sequence (see Section 3.5.2). In each case, we select the models by 4-fold cross-validation on the remaining 4 ADL sequences. Sequence details for this experiment are illustrated in Table 3.2. Tabulated data (features and labels) for this experiment is available here⁴.

- *Test performance:*

We present individual labeling results (F1-scores) from different models in Tables 3.12 (for locomotion activity) and 3.13 (for HL activity). The results for the joint labeling task are presented in Table 3.14.

Similar to the nested CV experiment on the UCI gesture phase data, FLDCRF-s improves the overall LDCRF performance on the dataset and outperforms LSTM and all other models. LSTM with the regular softmax layer for classification outperforms other LSTM models and LDCRF. LDCRF does not significantly improve CRF performance by modeling the latent dynamics, neither do the multi-label models FCRF, CCRF by considering joint learning. However, FLDCRF-m2 achieves notable improvement ($\sim 0.4\%$ on average) over these models by considering latent-dynamic interactions (cotemporal and first-order Markov, see Section

⁴<https://github.com/satyajitneogiju/FLDCRF-for-sequence-labeling>

3.6.2) among the label categories (locomotion and HL). It should be noted that, multiple hidden layers (1-3) for the same activity label were considered while computing results for FLDCRF-s. Therefore, multiple hidden layers for each label category can be employed in a FLDCRF-m2 model in order to further improve its performance. FLDCRF-m1 only includes co-temporal interaction among the hidden layers (see Section 3.6.2), and achieves marginal improvement over other CRF models.

Table 3.12: F1-scores for different models on the ‘locomotion’ activity label during nested CV experiment on UCI opportunity data.

Model	F1-score (Locomotion)					
	1	2	3	4	5	Average
CRF	48.26	64.4	84.01	82.32	86.87	73.17
LDCRF	48.3	64.4	84.04	82.32	86.87	73.19
FLDCRF-s	48.3	64.4	84.04	85.83	86.87	73.89
FCRF	48.16	64.42	83.93	82.53	86.84	73.18
CCRF	48.16	64.32	83.95	82.56	86.74	73.15
FLDCRF-m1	48.23	64.4	83.98	82.84	86.87	73.26
FLDCRF-m2	48.23	64.52	84.23	83.88	86.93	73.56
LSTM	46.41	64.28	77.64	91.73	87.94	73.60
LSTM-CRF	46.02	59.98	73.48	89.19	81.16	69.97
LSTM-m	46.60	61.47	78.66	90.63	88.38	73.15

Table 3.13 presents the labeling performance of different models on continuous tagging of the high-level (HL) activity. LDCRF, FLDCRF-s, FCRF and CCRF produce similar overall performance, marginally improving over the simple LC-CRF (or CRF) model. LSTM-CRF outperforms LSTM and LSTM-m models and achieves a similar overall performance to that of the LCCRF model. FLDCRF-m2, by considering the latent-dynamic interactions in the joint labeling task, significantly (by at least 0.75% on average) outperforms all CRF and LSTM models, with significant (>3%) improvement over LDCRF on set 1. As in case of locomotion, FLDCRF-m1 achieves marginal improvement over other CRF models.

Table 3.14 summarizes the joint labeling results of different models. FLDCRF-s outperforms all single-label sequence models, while FLDCRF-m2 outshines all single and multi-label sequence models. LSTM models fail to perform consistently

Table 3.13: F1-scores for different models on the ‘high-level’ activity (HL) label during nested CV experiment on UCI opportunity data.

Model	F1-score (HL)					
	1	2	3	4	5	Average
CRF	80.29	97.43	99.57	100	100	95.46
LDCRF	81.04	97.57	99.57	100	100	95.64
FLDCRF-s	81.04	97.55	99.56	100	100	95.63
FCRF	80.92	97.7	99.6	100	100	95.64
CCRF	80.91	97.71	99.57	100	100	95.64
FLDCRF-m1	81.38	97.73	99.72	99.88	100	95.74
FLDCRF-m2	84.29	98.1	99.56	100	100	96.39
LSTM	94.26	90.57	100	100	89.04	94.77
LSTM-CRF	81.21	98.04	100	100	98.02	95.45
LSTM-m	80.05	97.01	99.57	100	98.28	94.98

across the test sets, and yield smaller overall F1-scores. The multi-label LSTM-m achieves the best results among the LSTM models. FLDCRF-m2 improves the LDCRF, FCRF, CCRF and CRF performance on all sets, demonstrating the significance of modeling latent-dynamic interactions among different label categories. FLDCRF-m2 outshines the LSTM model on 3 (out of 5) sets, and outperforms LSTM-CRF and LSTM-m models on 4 sets each, while significantly outperforming (by $>1.5\%$) all LSTM models on the entire dataset.

- *Model selection:*

We present the validation performance of the best models from each family, FLDCRF-m (written for FLDCRF-m2) and LSTM-m, on cross-validation sets 2 and 5 in Tables 3.15 and 3.16 respectively. We omit the detailed results for other sets to avoid redundancy. We summarize the best-cases, worst-cases and standard deviations among hyperparameters during cross-validation on the inner loops in Table 3.17.

LSTM-m is associated with rapidly fluctuating cross-validation F1-scores across training epochs (N_{els}) for most of the considered N_{hls} settings (see Table 3.16), especially during cross-validation on set 5. It also produces significantly lower worst-case performances with high percentage of models giving notably poor results (19.5% of all models across 5 sets with $F1 < 72$, see Table 3.17). FLDCRF-m, on the other hand, shows much discipline against the choices of the hyperparameters,

Table 3.14: F1-scores for different models on the joint sequence labeling task during nested CV experiment on UCI opportunity data.

Model	F1-score (Overall)					
	1	2	3	4	5	Average
CRF	57.67	79.69	91.58	91.16	93.43	82.71
LDCRF	58.21	79.83	91.59	91.16	93.43	82.84
FLDCRF-s	58.21	79.80	91.58	92.92	93.45	83.19
FCRF	58.05	79.97	91.56	91.27	93.42	82.85
CCRF	58.05	79.93	91.55	91.28	93.37	82.84
FLDCRF-m1	58.42	79.98	91.71	91.3	93.43	82.97
FLDCRF-m2	60.53	80.4	91.67	91.94	93.47	83.60
LSTM	67.78	73.52	88.82	95.86	84.09	82.01
LSTM-CRF	57.19	78.07	86.74	94.60	88.64	81.05
LSTM-m	56.67	77.83	88.90	95.31	92.50	82.24

and shows decline in performance beyond certain N_h/N_s settings (see Tables 3.15 and 3.17), making it easier to optimize.

- *Consistency:*

The best LSTM models on the validation data outperform the best FLDCRF models during cross-validation for model selection (see Table 3.17). However, such optimized LSTM-m models fail to consistently beat FLDCRF-m on the test sets, instead they lead to inconsistent performance across test sets (yielding superior test performance only on 1 out of 5 sets, see Table 3.14). This is true for all other LSTM variants tested in this experiment.

- *Computation times:*

Figure 3.13 compares training times required by different FLDCRF-m and LSTM-m models. As mentioned earlier, FLDCRF-m is executed on a CPU and can be made faster by GPU implementation.

3.7.4 Multi-view Experiment

In this section, we examine the multi-view LDCRFs [74] (see Fig. 3.6) on the UCI opportunity data. We perform nested CV experiments separately on the locomotion

Table 3.15: FLDCRF-m2 validation performance on nested cross-validation sets 2 and 5 on UCI opportunity data.

Set 2									
FLDCRF	1/1	1/2	1/3	1/4	1/5	1/6	Best	Worst	Std
F1	76.17	76.18	76.23	76.2	76.63	76.2	76.84	75.34	0.33
FLDCRF	2/1	2/2	2/3	2/4	2/5	2/6			
F1	76.32	76.84	76.33	76.51	76.32	76.32			
FLDCRF	3/1	3/2	3/3						
F1	76.31	76.65	75.34						
Set 5									
FLDCRF	1/1	1/2	1/3	1/4	1/5	1/6	Best	Worst	Std
F1	73.27	74.48	73.29	73.27	74.37	73.62	74.48	72.78	0.46
FLDCRF	2/1	2/2	2/3	2/4	2/5	2/6			
F1	73.32	73.36	74.11	73.3	73.3	73.53			
FLDCRF	3/1	3/2	3/3						
F1	73.33	73.31	72.78

Figure 3.13: Average training time required by FLDCRF-m and LSTM-m per outer fold of UCI Opportunity data.

and HL activity labels of the UCI opportunity data. As described in Section 3.5.2, the feature set x_t for opportunity data comprises of 113 body-worn sensor features ($x_{1,t}$) and 32 shoe sensor ($x_{2,t}$) features i.e., $x_t = \{x_{1,t}, x_{2,t}\}$. We consider three different LDCRFs (with observations $x_{1,t}$, $x_{2,t}$ and x_t respectively, see Fig. 3.1c) and the best performing MVLDLCRF (among linked, coupled and linked-coupled variants [74]) for comparison. We also consider the FLDCRF-s performance with x_t . We present results of compared models on 10 test sets in Table 3.18.

Table 3.16: LSTM-m validation performance on nested cross-validation sets 2 and 5 on UCI opportunity data.

Set 2											
$N_{hls} \backslash N_{els}$	5	10	25	50	75	150	300	500	Best	Worst	Std
100	81.1	79.4	79.6	79.7	79.3	81.1	81.3	70.1	83.23	67.98	3.49
200	74.0	80.2	83.2	76.4	75.6	78.2	75.0	73.6			
300	80.5	77.6	81.8	75.5	80.3	78.2	80.8	72.5			
400	75.3	75.1	77.5	74.3	77.6	76.1	76.5	68.1			
500	77.6	75.9	77.2	76.2	74.7	76.1	75.9	67.9			
Set 5											
$N_{hls} \backslash N_{els}$	5	10	25	50	75	150	300	500	Best	Worst	Std
100	71.6	73.3	71.7	77.4	68.6	81.1	72.2	77.6	85.7	67.49	4.17
200	75.0	70.9	71.3	80.6	67.4	73.6	72.2	78.9			
300	78.3	71.1	70.1	71.4	70.0	70.1	69.2	79.6			
400	71.6	70.4	78.8	72.0	71.3	74	72.2	78.9			
500	72.1	71.4	73.2	72.3	78.8	85.7	70.8	79.6			

LDCRF-body, LDCRF-shoe and LDCRF-(body+shoe) corresponds to LDCRFs with observations $x_{1,t}$, $x_{2,t}$ and x_t respectively. LDCRF-averaged gives the mean of LDCRF-body and LDCRF-shoe performances. MVLDLDCF-averaged performance corresponds to the training and weighted inference mechanism by [74], while MVLDLDCF-body layer and MVLDLDCF-shoe layer gives the F1-scores obtained from individual layers of Fig. 3.6.

In most of the cases (sets 1, 3 of HL; sets 1, 5 of locomotion) where MVLDLDCF-averaged [74] outperformed the LDCRF with x_t , i.e., LDCRF-(body+shoe), either LDCRF-body (set 5 of locomotion) or LDCRF-shoe (sets 1, 3 of HL; set1 of locomotion) has outperformed all other models, including MVLDLDCF-averaged. In most other cases, one of the LDCRFs (LDCRF-(body+shoe), LDCRF-body, LDCRF-shoe) has outperformed MVLDLDCF-averaged. Therefore, it is necessary to compare all 3 LDCRF performances with MVLDLDCF-averaged, whereas Song et al. [74] only compared MVLDLDCF-averaged with LDCRF- x_t in their paper.

MVLDLDCF-averaged marginally outperforms all 3 LDCRF models (i.e., with

Table 3.17: Summary of cross-validation performance for FLDCRF-s and LSTM on the inner loops of nested CV experiment on the UCI opportunity data.

Model	F1-score (best)				
	1	2	3	4	5
FLDCRF-m2	87.54	76.84	75.12	74.31	74.48
LSTM-m	87.70	83.23	79.45	80.15	85.70
Model	F1 score (worst)				
	1	2	3	4	5
FLDCRF-m2	86.96	75.34	73.25	72.67	72.78
LSTM-m	79.26	67.98	68.81	65.23	67.49
Model	F1 score (std)				
	1	2	3	4	5
FLDCRF-m2	0.17	0.33	0.66	0.45	0.46
LSTM-m	2.00	3.49	3.35	4.26	4.17
Model	% F1 < 72				
	1	2	3	4	5
FLDCRF-m2	0	0	0	0	0
LSTM-m	0	10	17.5	27.5	42.5

$x_{1,t}$, $x_{2,t}$ and x_t) only on set 2 of HL, where all LDCRF-body, LDCRF-shoe and LDCRF-(body+shoe) have similar performance. We thus argue that an improvement by a MVLDCRF-averaged [74] over all three LDCRFs (with $x_{1,t}$, $x_{2,t}$ and x_t) is possible only when all 3 LDCRFs with $x_{1,t}$, $x_{2,t}$ and x_t perform similarly, as in set 2 of HL. Thus, it is advisable to test the three LDCRFs (or FLDCRF-s for improved performance) in case a feature distribution ($x_{1,t}$ and $x_{2,t}$) is available, else to utilize the entire feature set x_t with a FLDCRF-s and let the model learn the different interacting latent dynamics within the features and labels. Figuring out the distributions ($x_{1,t}$, $x_{2,t}$ etc.), where all 3 LDCRFs perform similarly is potentially one way for a MVLDCRF to perform better than all three LDCRFs (and possibly FLDCRF-s). However such a task is quite tedious, if not impossible.

Table 3.18: Comparing different models by distributing body-worn (113) and shoe sensor (32) features. LDCRF-body, LDCRF-shoe and LDCRF-(body+shoe) corresponds to $x_{1,t}$, $x_{2,t}$ and x_t respectively. LDCRF-averaged gives the mean of LDCRF-body and LDCRF-shoe outputs. MVLDCRF-averaged performance corresponds to the training and weighted inference mechanism by [74], while MVLDCRF-body layer and MVLDCRF-shoe layer gives the F1-scores obtained from individual layers of Fig. 3.6.

Model	HL					
	Set1	Set2	Set3	Set4	Set5	Average
LDCRF-body	79.96	97.13	99.57	99.83	100	95.29
LDCRF-shoe	98.48	97.83	99.81	99.97	99.94	99.21
LDCRF-averaged	89.22	97.48	99.69	99.9	99.97	97.25
LDCRF-(body+shoe)	81.04	97.57	99.57	100	100	95.64
FLDCRF-s(body+shoe)	81.04	97.55	99.56	100	100	95.63
MVLDCRF-body layer	86.63	97.96	99.63	100	100	96.84
MVLDCRF-shoe layer	91.81	98.18	99.71	100	100	97.94
MVLDCRF-averaged	88.01	98.14	99.68	100	100	97.39

Model	Locomotion					
	Set1	Set2	Set3	Set4	Set5	Average
LDCRF-body	46.6	60.33	81.5	82.9	87.22	71.71
LDCRF-shoe	51.55	71.97	76.81	81.34	77.81	71.9
LDCRF-averaged	49.07	66.15	79.155	82.12	82.51	71.80
LDCRF-(body+shoe)	48.3	64.4	84.04	82.32	86.87	73.19
FLDCRF-s(body+shoe)	48.3	64.4	84.04	85.83	86.87	73.89
MVLDCRF-body layer	48.26	63.93	83.98	81.95	87.09	73.04
MVLDCRF-shoe layer	45.92	63.58	83.98	82.05	86.99	72.50
MVLDCRF-averaged	48.43	64.40	83.98	81.95	87.03	73.18

3.8 Discussion

We have shown difficulties to select LSTM models on validation data across experiments in the paper. LSTM models, although producing excellent performance on validation data for certain hyperparameter settings, fail to perform consistently on the test data. FLDCRF outperforms LSTM on test data across all our experiments. Additionally, FLDCRF requires less computation times for training and inference,

even without GPU implementation. Moreover, since FLDCRF is a graphical model, it is straightforward to include known dependency information among the variables for improved modeling. We summarize the modeling attributes of FLDCRF and LSTM in Table 3.19, in terms of user convenience, as validated by experiments in this thesis.

Table 3.19: Comparison of FLDCRF and LSTM modeling attributes. ‘++’ stands for very convenient for users, ‘+’ stands for convenient, ‘-’ stands for inconvenient and ‘- -’ stands for very inconvenient.

Attribute	Model	
	FLDCRF	LSTM
Test performance	+	-
Ease of model selection	+	- -
• <i>Fewer hyperparameters</i>	+	-
• <i>Rules to select hyperparameters</i>	-	-
• <i>Tune training epochs</i>	++	- -
• <i>Patterns among hyperparameters</i>	++	- -
• <i>Worst-case validation performance</i>	+	- -
• <i>% of poor models</i>	+	- -
• <i>Variance among models</i>	+	- -
• <i>Best-case validation performance</i>	-	+
Consistency across validation and test sets	++	- -
Computation time	+	-
Easy to incorporate known dependencies	++	-
Intuitive	+	-
Practical deployment	+	-

Training and test data utilized in this chapter are of sizes in the $\mathcal{O}(1,000)$ and $\mathcal{O}(10,000)$ instances (see Section 3.5). We show FLDCRF to outperform LSTM on another three different but similar sized datasets in Chapters 4 and 5 [1]. Therefore, we can potentially argue that due to the better performance on the test data and the consistency across validation and test data, the FLDCRF model is more suitable for practical deployment than LSTM models on datasets of this scale.

However, LSTM models are particularly known to be successful with massive

datasets, i.e., data with $\mathcal{O}(100,000)$ instances, e.g., CNN-LSTM models [29] for action recognition on UCF-101 [94] and HMDB-51 [95] datasets, a BiLSTM-CRF model [53] for CoNLL 2000 chunking [71] task etc. By varying the number of hidden layers N_h and the number of hidden states associated to class labels in each layer N_{si} , $i = 1 : L$ (see Section 3.4), FLDCRF has the potential to be fitted efficiently to datasets with variable sizes. As shown in our numerical experiments (see Fig. 3.12), such extensions with additional layers and states are more computationally complex and would potentially require GPU implementations. We need to compare FLDCRF and LSTM models on larger sequence datasets [71, 94, 95] in order to generalize the aforementioned statement regarding practical deployment to all datasets.

FLDCRF subsumes well-known sequential CRF variants, viz., LCCRF, LDCRF and DCRF (see Section 3.4) and outperforms each of these state-of-the-art models across experiments in the paper.

In this thesis, we have utilized simplified versions of FLDCRF. For example, we have assumed first-order Markov dependency among different hidden layers. We also employed only one hidden layer per label category in the FLDCRF-m models for simplicity. Furthermore, we did not consider past input features (x_{t-1} , x_{t-2} etc.) during modeling. A general FLDCRF structure (see Fig. 3.7 and expression (3.5)), allows more generic FLDCRF variants. Such more complex variants can be obtained by simply modifying the feature functions of (3.8) – (3.10), and can be exploited for even better performance, depending on the system capabilities (RAM, CPU etc.) and available training data.

We also did not consider future state/label variables (y_{t+1} , y_{t+2} etc.) and future inputs (x_{t+1} , x_{t+2} etc.) during training and inference for all considered models. All models (FLDCRF and LSTM) can be enhanced by such considerations, e.g., a BiLSTM-CRF model [53], forward-backward inference [46] for FLDCRF etc. However, such inference mechanisms are only applicable after observing future inputs, i.e., for smoothing, when processing delays are tolerable.

3.9 Conclusion

We proposed FLDCRF, a single and multi-label generalization of LDCRF. We presented 3 different FLDCRF variants for single-label (FLDCRF-s), multi-label (FLDCRF-m) and multi-agent (FLDCRF-i) sequence prediction/tagging tasks. FLDCRF-s allows multiple interacting latent dynamics of the class labels and extends the capability of LDCRF, as well as outperforms LSTM and LSTM-CRF across multiple datasets. FLDCRF-m introduces hidden variables in a DCRF to accommodate la-

tent dynamic interactions among different label categories, thereby improving DCRF performance and outperforming all state-of-the-art sequence models including CRF, LDCRF, FCRF, LSTM, LSTM-m on a joint sequence labeling task. We also observed difficulties associated with optimizing LSTM models and the inconsistent performance across validation and test data (summarized in Table 3.19) for LSTM models. By contrast, FLDCRF presents easier model selection and provides consistency across validation and test data. FLDCRF also offers lucid model intuition and user flexibility. We plan to design GPU implementation of FLDCRF and compare FLDCRF and LSTM on larger sequence datasets. Another interesting topic for future research is to apply end-to-end models with CNN layers followed by FLDCRF (i.e., a CNN-FLDCRF) on popular computer vision problems like action recognition [29]. We are also exploring the multi-agent FLDCRF-i model for joint intention prediction of pedestrians [1] for applications in the fields of Autonomous Vehicles and Advanced Driver Assistance Systems. It is also possible to extend the idea of factorized latent space in FLDCRF to heterogeneous (discrete and continuous) state space models, which we will explore in future work. We describe such possible extensions and other future research on FLDCRF in detail in Chapter 6.

Next, we apply FLDCRF for pedestrian intention prediction in Chapters 4 and 5.

Chapter 4

Dynamic Context Model for Pedestrian Intention Prediction using FLDCRF

Smooth pedestrian interactions is still an area of active research for Autonomous Vehicles (AV) and Advanced Driver Assistance Systems (ADAS). A human driver may require assistance with stable and early prediction of pedestrian crossing behaviours while navigating through a crowded scene, or may need a system to smoothly handle crossing/not-crossing pedestrians while being absent-minded. A safe and smooth autonomous interaction with pedestrians should not be very conservative, where the vehicle stops to give way to even a not-crossing pedestrian. At the same time, the system should avoid emergency hard braking as much as possible. Ideally, there should be sufficient path-planning time for the system to achieve a smooth control over the vehicle dynamics.

In this chapter, we propose a joint (context+motion) model for stable and early pedestrian behaviour prediction and validate the model on two relatively simpler pedestrian datasets, viz., the most popular (for testing pedestrian behaviour prediction) and publicly available Daimler dataset [37] and an in-house collected NTU¹ dataset. We extend our approach to complex real-life pedestrian behaviours with two more datasets, viz., publicly available JAAD (Joint Attention for Autonomous Driving, [23]) data and Little India data² [5], in Chapter 5. Although FLDCRF outperformed LSTM across experiments and presented superior modeling attributes in Chapter 3, we apply and compare both FLDCRF and LSTM as our sequence models in all our subsequent experiments (in Chapters 4 and 5) in order to measure

¹derived from Nanyang Technological University.

²collected by a joint effort from ST Engineering and our research group.

robustness across datasets and applications.

4.1 Challenges

Pedestrian behaviour is the most random [93] and variable factor for any vehicle, with the situation varying from sparse to densely crowded areas. While 100% pedestrian safety is utmost priority, the vehicle also should ensure a smooth vehicle dynamics during navigation. Let us have a look at the existing challenges involving pedestrians on road for human drivers and AVs, and how it leads to the current trade-off between pedestrian safety and smoothness of an autonomous ride.

4.1.1 Is (s)he going to cross?

Figure 4.1a-d illustrates four day-to-day pedestrian encounters for a vehicle. There is always a task for a human driver to estimate whether the pedestrian(s) is/are going to cross in front of the vehicle or not, and take necessary action to ensure pedestrian safety. A human driver relies on the pedestrian motion, location, his/her distance from the vehicle, the vehicle speed, traffic elements such as bus stop, zebra crossing, traffic light etc., and other factors³ to make an estimate of the pedestrian behaviour. All situations involving pedestrian interactions demand optimum diligence and attention from a human driver. Several studies [89, 90] reflect that a staggering 44% of all road accidents occur due to an inattentive driver. An autonomous system, employed to predict pedestrian behaviour, can certainly obviate such fatalities caused due to absent-mindedness.

4.1.2 Who is going to cross?

The situation gets worse in the crowded scenes, as depicted in Fig. 4.2, and demands absolute attention from a human driver to avoid colliding with a crossing pedestrian. An autonomous system for stable and early indication of the crossing pedestrian(s) can certainly aid the driving process.

4.1.3 Safe ride or smooth ride?

Autonomous systems (AV and ADAS) promise to alleviate human errors and ensure better safety models for on-road pedestrians. However, these systems suffer from

³such as weather, scene type, e.g., crowded/not crowded etc., group of pedestrian/solo pedestrian etc.

Figure 4.1: Is he/she going to cross? Figures taken from the JAAD dataset [23].

Figure 4.2: Who is going to cross? Figure drawn from a crowded scene in the Little India data [5]. A pre-trained Faster R-CNN [34] is applied to obtain the pedestrian and vehicle bounding box detections.

the trade-off between on-road pedestrian safety and smoothness of the ride. Recent user experiences and available online footage suggest either conservative autonomous rides resulting from the emphasis on on-road pedestrian safety (the Waymo model, see Fig. 4.3 [91]), or aggressive rides resulting in fatal accidents (the Uber crash, see Fig. 4.3 [92]). To achieve rapid user adoption, the AVs must be able to simulate a smooth human driver-like experience without unnecessary interruptions, in addition to ensuring 100% pedestrian safety.

Automated braking systems in an ADAS tackle the emergency pedestrian interactions. These brakes get activated on detecting pedestrians' crossing behaviours within the vehicle safety range. Such a system may need to brake strongly on late prediction of pedestrian crossing behaviour, which can occur frequently in crowded areas. A future ADAS must be able to offer a smoother experience on such interac-

Figure 4.3: Safe ride or smooth ride? The AV trade-off [91, 92].

tions.

4.2 Proposed Solution

The key to a safe and smooth autonomous pedestrian interaction lies in early and accurate prediction of a pedestrian’s crossing/not-crossing behaviour in front of the vehicle. Accurate and timely prediction of pedestrian behaviour ensures on-road pedestrian safety, while early anticipation of the crossing/not-crossing behaviour offers more path planning time and consequently a smoother control over the vehicle dynamics.

A stable, early and accurate prediction system of a pedestrian crossing behaviour can also assist absent-minded drivers through visual aids and alarms, thereby ensuring safer streets.

4.2.1 The objective statement

Since each pedestrian interaction with the ego-vehicle⁴ is continuous in time for a certain duration T_p and we want to classify crossing/not-crossing intentions of a pedestrian at each instant within this duration, we formulate this problem as a continuous online sequence labeling task (see Section 2.2. in Chapter 2) given by:

$$\hat{y}_t = \operatorname{argmax}_{y_t} P(y_t \mid x_{1:t}), \quad t \in 1: T_p, \quad (4.1)$$

where y_t is the intention variable to be predicted and x_t are the input features to the system at t . $y_t \in \{\text{‘crossing’}, \text{‘not-crossing’}\}$.

⁴the vehicle concerned with the predictions is termed as a ego-vehicle.

4.3 Literature Review

From an Autonomous Vehicle perspective, there are two popular problems concerning pedestrian behaviour predictions, a) continuous path prediction [7], [18] - [22], [24], [25], [26], [27], [28], [37] where future trajectory of a pedestrian is estimated and b) discrete intention prediction [7] - [17], where the goal is to predict discretized pedestrian behaviours. Given a detected pedestrian, his/her position w.r.t. the AV and a marked AV safety range, solving any problem (path or intent prediction) can ensure a safe autonomous ride. However, a smooth autonomous ride of (avoiding repeated halts and hard brakes) would require early predictions of pedestrian behaviour for optimal path and dynamics planning. Although pedestrian interactions and static traffic scene elements have been successfully utilized [20] - [22], [24], [25], [26], [27] to improve path prediction accuracy, long-term path prediction (>1 sec ahead) is still challenging and error prone, with the prediction error growing exponentially with prediction horizon [20]. Additionally, path prediction given fewer (≤ 10) observed frames, distant objects and moving cameras invokes additional uncertainties. In this thesis, we analyze the feasibility of accurate long-term (up to 2 sec ahead) discrete intent prediction in a variety of scenarios.

Existing intent prediction approaches can be broadly classified into two categories, a) Pedestrian-only, which only take features from the pedestrian of interest and b) Context based, which consider context variables (location, traffic elements etc.) together with pedestrian features.

4.3.1 Pedestrian-only approaches

Pedestrian-only approaches [7], [9], [10], [13], [16], [17], in general, rely on machine learning models over pedestrian features (motion and/or appearance) and n-ary intention labels. Pedestrian features include dense optical flow [7], concatenated position, velocity and head pose [9, 18], Motion Contour image based Histogram of Oriented Gradients descriptor (MCHOG) [10], skeletal features [13], [16] and deep learning appearance features [17]. Applied machine learning have been either time-series models (GPDM, PHTM [7], LDCRF [9], HMM [13], LSTM [17]) or SVM [10], [16].

4.3.2 Context based approaches

Context based intent prediction approaches [8], [11], [12], [14] consider information like scene spatial layout, i.e. location of curb [8], [11] and traffic elements [8], [11],

[12], [14], such as cross walks, bus stops, traffic lights, zebra crossings etc. along with pedestrian features.

The majority of these approaches [7], [9], [16], [17] were evaluated on the Daimler dataset [37], a public benchmark consisting of pedestrian crossing/not-crossing sequences. Stopping probability vs time [7], [9], [16], [17] and classification accuracy [7], [10], [13], [14] are the most common metrics. Although the Daimler data is very popular, it is not real-life and it only involves acting pedestrians and an acting ego-vehicle. At the same time, the ego-vehicle interaction with the pedestrian is not made to be spontaneous. Hence, we utilize this dataset only to validate our pedestrian motion model. We augment this motion-only model with vehicle (and location) context for application on the NTU dataset (see Section 4.6.2). We compare our pedestrian motion-only approach with the state-of-the-art model [7] on the Daimler dataset in Section 4.9.2.

Recently a group of researchers have proposed a relatively large, labeled pedestrian dataset called JAAD [23], captured from moving vehicles. This dataset provides real-life examples of pedestrian interactions with a vehicle in complex scenes. In [14], the same group proposes a context based system to classify pedestrian crossing/not-crossing behaviour, where they combine an action recognition classifier and a learned attention model for relevant static context variables (traffic lights, zebra crossings, stop signs etc.) to classify pedestrian intention. In [15], Gujjar et al. showed improvements in performance on the same dataset by considering a future generation model. However, we find their evaluation dataset to be weakly relevant to early prediction of intention (evaluation does not involve any temporal information) and hence select our own evaluation sequences (described in Chapter 5.3.1) from the JAAD dataset. We apply our intention prediction model on the JAAD dataset in Chapter 5.

4.4 Proposed Intention Prediction Model

As pedestrians, our crossing/not-crossing intentions are largely influenced by the distances and speeds of the approaching vehicles. We refer to this causal information as dynamic context or *vehicle interaction context* (see Fig. 4.4) and utilize it together with pedestrian motion and location information for early prediction of future behaviour. To the best of our knowledge, we are the first [3] to consider influence of such dynamic context on pedestrian intention, which proves to be the strongest factor for early prediction, as we demonstrate in Section 4.9. Adding such context improves prediction time of the pedestrian stopping (at the curb) behaviour by ≥ 0.5 seconds on average across considered datasets in this chapter (see Section 4.9) and in

Figure 4.4: Overview of the proposed intention prediction system. Points in red highlight our main contributions. Figure taken from [1].

Chapter 5, and can potentially reduce unnecessary halts in future autonomous rides. This dynamic context also improves prediction time and accuracy of other behaviour types (e.g., crossing) across the datasets (see Section 4.9).

Existing literature on context models for pedestrian behaviour prediction (see Section 4.3) rely on static scene context variables and do not consider dynamic vehicle interactions. While [13] presents a 70% prediction accuracy of pedestrian stopping actions 0.06 seconds before the event on [38], we achieve such accuracy at least 0.9 seconds before the event across our datasets in presence of vehicle interactions. Current best approach [17] on the Daimler dataset gives such an accuracy 0.38 seconds before event, although it is tested on a small test dataset (with 13 pedestrian sequences) with limited variations in pedestrian behaviour.

In addition to early and accurate pedestrian intention prediction, we put stress on the necessity to simulate real-life prediction environment for our experiments. We apply state-of-the-art object detectors [34] and semantic segmentation systems [35] to detect pedestrians/vehicles and to determine pedestrian location respectively. Fig. 4.4 outlines the different modules and their inter-connections in our intention prediction system. We apply the proposed single label variant of FLDCRF (FLDCRF-s, see Chapter 3.4) and compare with LSTM over input features (context + motion) and intention labels (see Section 4.8.2).

4.5 Nomenclature

In general, a pedestrian’s crossing/not-crossing behaviour in front of the ego-vehicle is characterized by the real world lateral component of his/her motion w.r.t. the ego-vehicle moving direction, i.e.,

$$\begin{aligned}
\textit{laterally moving} &\longleftrightarrow \textit{crossing} \\
\textit{laterally static} &\longleftrightarrow \textit{not - crossing}.
\end{aligned}$$

At each instant t , we try to predict whether a detected pedestrian will be laterally moving or laterally static⁵ in the near future. Associated critical measures and necessary vehicle dynamics can be derived by further analyzing pedestrian’s position w.r.t. the vehicle.

We evaluate pedestrian sequences where early prediction is relevant from an AV perspective. Inspired by the work in [7], in this chapter we consider the scenarios of a crossing pedestrian, who may cross or stop before the ego-vehicle (see Fig. 4.5). Thus, considered pedestrian sequences for evaluation in this chapter⁶ are of two types:

1. **Continuous crossing:** The pedestrian initially has a lateral movement before the curb, continues to move laterally and crosses in front of the vehicle.
2. **Stopping at the curb:** The pedestrian initially has a lateral movement before the curb and stops at/near the curb. We call this behaviour ‘stopping’ for simplicity.

Stopping sequences carry a change in state of motion (laterally moving to laterally static) of the pedestrian. As our goal is to predict pedestrian intention early, our primary aim is to detect these changes of state early (preferably before any discernible change in motion/appearance), accurately and with stability. In addition, we also want stable and accurate predictions of the ‘crossing’ intention for continuous crossing sequences.

Each sequence type is associated with an *event instant* (e.g., *crossing*, *stopping*), illustrated in Fig. 4.5, which provides the ground truth classification labels and also serves as reference during evaluation. To aid the labeling process during training (for early prediction of the transitions in stopping sequences), we define a *pred_ahead* parameter, in number of frames before the respective event instants. We explain how we labeled the pedestrian sequences during training in Section 4.8.

4.6 Datasets

We evaluate our models on two datasets in this Chapter: 1) Daimler Dataset [37] and 2) NTU Dataset [4]. We briefly discuss the dataset characteristics below. The idea is

⁵Laterally static instances include pedestrians walking along the curb or sidewalk.

⁶Realizing more possible types of pedestrian behaviours in real-life data, we propose a revised nomenclature in next chapter (see Chapter 5.2.2) to encode majority of encountered pedestrian behaviour types.

Figure 4.5: Types of pedestrian behaviours considered in this chapter. Figures drawn from the JAAD [23] dataset. EI stands for event instant (crossing, stopping etc.).

to verify our approach on simpler datasets, and then extend our approach to real-life scenarios in Chapter 5. We validate our proposed motion features (see Section 4.7.2) on the Daimler dataset, and experiment on our joint (context + motion) approach on the NTU data [1, 4].

4.6.1 Daimler Dataset

The Daimler dataset is a public pedestrian benchmark introduced in [37]. It contains several single pedestrian movement sequences captured at 16 fps by a stereo camera setup mounted on a moving vehicle. The movements primarily include three distinct scenarios: a) A laterally moving pedestrian ‘crossing’ or ‘stopping’ before a vehicle and b) A pedestrian moving along the sidewalk ‘bending in’ to the road and c) A pedestrian ‘starting’ to cross the road in front of a vehicle from a static position. Ground truth labels are provided by means of bounding boxes. For all scenarios, important events like ‘stopping’, ‘crossing’, ‘bending’ and ‘starting’ instants are marked as 0-th instant. The whole dataset contains 36 training and 32 testing scenarios. The number of sequences under each category is tabulated in Table 4.1. Each video sequence has a resolution 1176×640 .

Table 4.1: Table showing distribution of sequences in Daimler dataset

Type	Crossing	Stopping	Starting	Bending in
Training	8	9	10	9
Testing	5	8	10	9

4.6.2 In-house NTU Data

The Daimler dataset does not involve real scenarios with on-road vehicles influencing pedestrian crossing/not-crossing behaviours. Therefore, we collect an in-house NTU dataset, in order to capture the influence of approaching vehicles on pedestrian intentions.

We validate our joint (context+motion) models (see Fig. 4.4) for intention prediction on the NTU dataset [1] in this chapter. In Chapter 5, we extend our approach to real-life datasets, viz., JAAD dataset [23] and Little India data [5].

The NTU dataset is captured inside Nanyang Technological University Campus by a pair of static cameras at 30 fps frame rate and 1920×1080 resolution. Since our aim is to capture natural vehicle influences on pedestrian intention, we place two synchronized cameras to capture the whole scene of interest, so that approaching vehicles on both sides of the pedestrian can be captured, as depicted in Fig. 4.6. In addition, one of the cameras (Camera_1) is placed in a way to simulate a camera mounted on a stationery ego-vehicle.

Figure 4.6: NTU dataset sample. Camera_1 (left) and Camera_2 (right) together capture the whole scene of interest.

In the videos, we had actors and actual pedestrians (age group 20-28) strolling in a natural road scene with vehicles. Actors were not given any specific instructions and were asked to move around the scene naturally. We extract 35 continuous crossing and 35 stopping sequences from the videos for our evaluation.

4.7 Feature Extraction

We describe here the process to prepare our input feature vector x_t for the models. x_t comprises of two components: context features, $x_{c,t}$, and motion features, $x_{m,t}$.

4.7.1 Context Features

We consider two types of context features:

Figure 4.7: Feature extraction: (a)-(c) depict pedestrian location context. Semantic segmentation of the scene in (a) is presented in (b). Road pixels are highlighted. (c) shows the different regions around pedestrian bounding box to capture necessary location context variables. (d) highlights pre-calibrated depth lines. Figures taken from the NTU dataset. (e) 9 points are selected symmetrically within a pedestrian bounding box during motion feature extraction.

- **Pedestrian location context:** Also called *spatial context* in the paper, this context variable encodes the location (e.g., on road, at curb, away) of the pedestrian in a scene.
- **Vehicle interaction context:** We call it *vehicle context* in the paper. This context variable contains approximate longitudinal distance (depth) and velocity measures of the nearest vehicles to the pedestrian of interest.

Pedestrian location context

Fig. 4.7(a-b) show a scene and its semantic segmentation. We utilize two segmentation categories, road pixels and non-road pixels. Road pixels are directly obtained from the pre-trained semantic segmentation software [35], and all other pixels are marked as non-road pixels. We highlight 3 different regions around the pedestrian bounding box and their considered dimensions in Fig. 4.7c. We compute the mode of the refined semantic values (road or non-road) in each of the three regions, given by $\{m_1, m_2, m_3\}$, which constitute the location context feature vector $x_{sp,t}$.

Vehicle interaction context

Longitudinal distances (or depths) and velocities of approaching vehicles play a significant role in determining crossing intention of a pedestrian.

Depth estimation: As existing techniques for monocular depth estimation are not very reliable, we employ a pre-calibrated approach.

To estimate pedestrian and vehicle depths from camera, we pre-calibrate a scene by 10m, 20m and 30m depth lines (see Fig. 4.7d). We assume that once a camera is steadily mounted on a vehicle, these depth lines will approximately remain the same during driving, unless the road ahead has high slope⁷. The depth lines can be written as:

$$y = mx + c, \quad (4.2)$$

where (x, y) is a point co-ordinate on the image (in pixels). We assume that the slope m and the intercept c (in pixels) of the straight line have the following approximate exponential⁸ relations with depth d (in meters) from the camera:

$$m(d) = k_m - p_m \cdot e^{-dl_m}, \quad (4.3)$$

$$c(d) = k_c - p_c \cdot e^{-dl_c}, \quad (4.4)$$

where $m(d)$ and $c(d)$ are known for $d = 10, 20$ and 30 . The parameters k_m, p_m, l_m, k_c, p_c and l_c can be determined by solving equations (4.3) and (4.4) with these 3 pairs of values of $\{m(d), c(d), d\}$. Once these parameters are obtained, depth of any given point (x, y) in the image can be approximately determined from (4.2), (4.3) and (4.4). We consider the bottom center pixels of the concerned pedestrian and vehicle bounding boxes to estimate their depths.

Velocity estimation: To obtain velocity measures of the nearest vehicles to a pedestrian in the NTU dataset, we compute mean optical flow [32] magnitude inside the corresponding vehicle bounding boxes. In the two-dimensional images, at times there are overlapping pixels between pedestrians and vehicles. Therefore, it is necessary to compute the mean optical flow within a vehicle bounding box after removing such overlapping pixels, in order to avoid misleading velocity estimates.

The complete (depth and velocity) vehicle context for the NTU dataset is given by, $x_{vNTU,t} = \{ncl, ncr, nclf, ncrf\}$, where ncl and ncr are the relative depths (in

⁷The depth lines may vary with tyre pressure. Since we only need approximate depths, our method worked well across datasets. For precise depth estimation, other techniques should be explored.

⁸The slope and intercept exponentially decay to a constant at the horizon.

meters) of the nearest vehicles to a pedestrian on left and right lane respectively and $nclf$, $ncrf$ are the mean optical flow of the *non-pedestrian* pixels within the bounding boxes of the respective vehicles. At instants with no influencing vehicle, we set ncl and ncr to -1 and $nclf$ and $ncrf$ to 0.

As described earlier, our model on the Daimler dataset only involves pedestrian motion features (see Section 4.7.2) and we do not utilize (spatial and vehicle) context on this dataset.

4.7.2 Motion Features

We propose a computationally inexpensive method to capture relevant pedestrian motion dynamics. The features obtained by this method perform comparably to the state-of-the-art [7] performance on the Daimler dataset [37] (see Section 4.9.2). We discuss the pedestrian motion feature extraction process below.

First, we apply a pre-trained object detector [34] to detect and track pedestrians in the videos⁹. For extracting pedestrian motion features at t , we consider a sliding window of length τ , i.e., frames $t - \tau + 1$ to t . The feature extraction process involves three steps:

- **Pre-processing:** We run a Kalman filter on tracked pedestrian bounding boxes to remove noise/occlusions. Further occlusions (e.g., by tree) are linearly interpolated.

- **Real world lateral displacement estimate:** As mentioned in Section 4.5, a pedestrian’s crossing/not-crossing behaviour is characterized by the real world lateral component of his/her motion w.r.t. the ego-vehicle moving direction. Lateral motion captured in an image sequence contains components from both lateral and longitudinal pedestrian movements (w.r.t. the ego-moving direction) in real world (see Fig. 4.8). In addition, the captured real world longitudinal movement is mixed with the ego-vehicle motion. Therefore, we need to extract the real world lateral component (along x_R) of the pedestrian motion in order to adequately represent the crossing/not-crossing behaviour, as well as to eliminate spurious longitudinal components.

Assuming negligible pedestrian movement along the vertical direction (z_R) in real world, we obtain approximate estimates of the real world lateral motion (S_{xx}) in image frame from the camera matrix information, as follows.

Fig. 4.8 depicts the scenario. O is the camera center and I_p is the principle point on the image plane. x_R and y_R are the lateral and longitudinal (w.r.t. the movement

⁹For considered pedestrians in the NTU data. We utilized provided bounding box annotations for pedestrians in the Daimler data.

Figure 4.8: Displacement in real world and its image equivalent. z_i image axis and z_R real world axis are pointed outwards w.r.t. the paper plane. y_R gives the ego-vehicle moving direction, i.e., the longitudinal direction.

direction of the ego-vehicle) directions in real world respectively. $\vec{PQ} = \vec{S}$ represents relative displacement of a pedestrian point w.r.t. the ego-vehicle in real world between two consecutive time instants. We assume negligible pedestrian displacement along the vertical real world direction z_R . As we can see, corresponding displacement along x_i image axis is S_{ix} , which contains components from both real world lateral (S_x) and longitudinal displacements (S_y), projected as S_{xx} and S_{yx} respectively onto the image plane, i.e., $S_{ix} = S_{xx} + S_{yx}$. Similarly, considering displacement along z_i image axis to be S_{iz} , we can write the following two equations:

$$\begin{aligned} S_{ix} &= \pm r_x \cdot \theta_x \pm r_x \cdot \theta_y, \\ S_{iz} &= \pm r_z \cdot \theta_y, \end{aligned} \quad (4.5)$$

Since we assume negligible pedestrian motion along vertical real world direction, S_{iz} only carries traces of pedestrian's real world longitudinal motion. $\pm$ indicates the components can be additive or subtractive depending on the image quadrant of movement. Considering $+\theta_x$ and $+\theta_y$ to reflect real world +ve x and +ve y direction respectively, different possible scenarios are illustrated on the image plane $x_i z_i$ in Fig. 4.9. r_x and r_z are the known respective distances from the camera center O to the center of the displacements (situated at (p_x, p_z) on image plane) along axes x_i and z_i , given by,

$$r_x = \sqrt{f^2 + p_x^2}, \quad r_z = \sqrt{f^2 + p_z^2} \quad (4.6)$$

We consider the center of displacement between two consecutive frames to measure r_x and r_z , in order to minimize the error while approximating $s = r \cdot \theta$ (for $\vec{s} = r \cdot \theta \cdot \hat{\theta} + dr \cdot \hat{r}$), to reduce number of unknown variables in equation (4.5). The goal is to solve for θ_x and θ_y from the equations illustrated in Fig. 4.9. S_{ix} , S_{iz} , r_x and r_z are known. S_{ix} , S_{iz} are the displacements of a given pedestrian point (see Fig. 4.7e)

between consecutive frames along x_i and z_i respectively and are computed by simple subtraction.

Figure 4.9: Sign conventions on the image plane. θ_y is considered to be along +ve y axis is real world.

S_{xx} gives approximate pedestrian's real world lateral motion equivalent on the image plane and is obtained from $S_{xx} = \pm r_x \cdot \theta_x$.

• **Fitting non-linear dynamics:** The final step is to fit a non-linear dynamics to the processed lateral displacement values. We extract real world lateral displacement estimates for 9 symmetrical points inside the pedestrian bounding box, given by $\{v_p\}$, $p = 1 : 9$ (see Fig. 4.7e). The choice of the points is arbitrary, as long as they are well distributed within the bounding box (BB) and their positions are specified within the BB for tracking across frames. For simplicity, we considered 9 symmetrical points within the BB, as shown in Fig. 4.7e. Considering such multiple distributed points encodes a measure of the intrinsic BB dynamics (such as expansion and contraction) between frames, in addition to the extrinsic dynamics of the overall BB lateral movement. Each of these points is tracked and processed over the sliding window to generate $\tau - 1$ lateral displacement values v_{p,t_1} , $t - \tau + 2 \leq t_1 \leq t$. Finally, we fit a degree 2 curve to the values $\{v_{p,t-\tau+2:t}\}$, $p = 1 : 9$, by least-squares fitting:

$$v_{p,t_1} = a_{0p,t} + a_{1p,t}t_1 + a_{2p,t}t_1^2, \quad (4.7)$$

resulting in a 27 dimensional feature set, $x_{m,t} = \{\{a_{0p,t}\}_{p=1:9}, \{a_{1p,t}\}_{p=1:9}, \{a_{2p,t}\}_{p=1:9}\}$. τ must be large enough (≥ 4) for a good fitting but not too large to include unnecessary past frames in the current dynamics computation. We set $\tau = 10$, but for test sequences with less than 10 frames, we adjust τ accordingly to generate the motion features. We also tested optical flow features and visual representations as in another work from our group [17]. However, including such features resulted in poorer performance across datasets in this chapter and Chapter 5. Therefore, we do not include such features in this thesis.

4.8 Experimental Setup

We describe here our experimental data, extracted features and labels, and the model specifications.

4.8.1 Data for Evaluation

The distribution of test sequences in the two datasets is presented in Table 4.2.

We utilized the train and test splits on the Daimler data as in [7]. The distribution of test sequences is presented in Table 4.2.

We downsampled the NTU dataset sequences to 15 fps. We detect and track pedestrians by a pre-trained Faster R-CNN [34] in Camera_1 images. We also define pedestrian *spatial context* only in Camera_1 images. Vehicles are detected and tracked [34] in both Camera_1 and Camera_2 images. We eliminate ambiguity within the common regions of Camera_1 and Camera_2 by a pre-defined separator (marked by black lines, see Fig. 4.6). The object detector could efficiently detect pedestrians and vehicles up to 80 meters (at ~ 0.02 sec/frame with downsampled frame size 960×540) from the camera.

Table 4.2: Table showing distribution of test sequences in different datasets.

Seq type \ Dataset	Continuous crossing	Stopping	Total # instances
Daimler	5	8	2235
NTU	35	35	9276

4.8.2 Features and Labels

In our quest to predict the ‘not-crossing’ event in stopping sequences (see Fig. 4.5b) early, we follow the labeling scheme proposed in [7]. On the Daimler data, we label data from 20 frames (1.25 sec) before the *stopping instant* in stopping sequences as ‘not-crossing’ during training. Remaining frames in stopping sequences and all frames in continuous crossing sequences are labeled ‘crossing’. The annotations are depicted in Fig. 4.10. As described earlier, we only apply our motion based approach on the Daimler data, hence the input feature set utilized on the Daimler data is given by $x_{t,Daim} = x_{m,t}$.

In the NTU data, we try to predict the stopping event 30 frames (2 sec) before the *stopping instant*, i.e., $pred_ahead = 30$. Therefore, we label data from 30 frames before the *stopping instant* in stopping sequences as ‘not-crossing’. Similar to the Daimler data, remaining frames in stopping sequences and all frames in continuous crossing sequences are labeled ‘crossing’. The annotations are presented in Fig. 4.10. The complete feature set (see Section 4.7) on the NTU data is given by, $x_{t,NTU} = \{x_{m,t}, x_{sp,t}, x_{vNTU,t}\}$.

Figure 4.10: Labeling the training sequences for the Daimler and NTU data. ‘c’ denotes ‘crossing’ and ‘n’ denotes ‘not-crossing’ intention. For the Daimler data, $pred_ahead = 20$ (1.25 s at 16 fps). The NTU data follows similar labeling with $pred_ahead = 30$ (2 s at 15 fps).

4.8.3 Comparison of the Models

We utilized the FLDCRF single-label (FLDCRF-s) variant and LSTM as our learning models on the NTU data. We show results for three different systems on the NTU data:

- 1) Only pedestrian motion based, referred as mtn (or m) in the Results section. This system only utilizes pedestrian motion as input features, i.e., $x_t = \{x_{m,t}\}$.
- 2) Pedestrian motion and location based, referred as $mtn + spa$ (or ms). This system takes pedestrian’s semantic location (see Section 4.7.1) along with pedestrian motion as inputs, i.e., $x_t = \{x_{m,t}, x_{sp,t}\}$.
- 3) Pedestrian motion, location and vehicle context based, referred as $mtn + spa + veh$ (or msv). This system considers vehicle context features (see Section 4.7.1) together with pedestrian motion and location, i.e., $x_t = \{x_{m,t}, x_{sp,t}, x_{v,t}\}$.

The idea is to examine influence of each feature component on the prediction outcome.

On the NTU data, we employ nested cross-validation (with 5 outer folds and 4 inner folds) for selecting models and generating results. For selecting LSTM models, we tested the following number of hidden units (HU): 2, 5, 10, 20, 50, 150, 300, 500. We trained each model for 1000 epochs, saving model performances at every

100 epochs during validation. We fed all training sequences at each epoch at the rate of 1 sequence per batch.

An FLDCRF-s has two hyper-parameters: number of hidden layers N_h and number of hidden states per label $\{N_{si}\}$ along layers $i = 1 : L$. For simplicity, we keep same N_s along all layers, i.e., $N_{si} = N_s, \forall i = 1 : L$. We denote such model by FLDCRF- N_h/N_s . We selected our models from the following N_h/N_s combinations: 1/1, 1/2, 1/3, 1/4, 1/5, 1/6, 2/1, 2/2, 2/3.

We trained the FLDCRF model parameters with Stan’s [31] in-built BFGS optimizer. We implemented the LSTM models defined in Keras, a deep learning library for Python running Tensorflow in the backend. LSTM parameters were trained by a default Adam optimizer in Keras. We plotted the results in MATLAB 2015b [36]. LSTM models are trained with a Nvidia Tesla K80 GPU, while the FLDCRF models are trained with a Intel Xeon E5-1630 v3 CPU.

As discussed earlier, Daimler data is not captured with spontaneous vehicle context, and thus we only utilize this data to validate our motion features. We compare a FLDCRF-1/4 trained with the motion features $x_{m,t}$ with the state-of-the-art performance [7] on this dataset.

4.9 Results

We compare the motion-only system with the state-of-the-art [7] model on the Daimler dataset, and establish our motion features. We compare all 3 different systems (see Section 4.8.3) on the NTU dataset using existing state-of-the-art (LSTM) and proposed (FLDCRF) sequential models. We evaluate the models by two metrics: stopping probability vs time and classification accuracy vs time. Stopping probability corresponds to the probability of the ‘not-crossing’ (laterally static) label.

4.9.1 Metric Definition

We ideally want high classification accuracy of appropriate class labels (e.g., ‘not-crossing’ in stopping sequences) over the prediction interval. At the same time, it is desired to have high and stable average probability values of appropriate class labels over the prediction interval. It is also preferred that the sequences of same type have minor standard deviation in predicted probability values at each instant. So, our model hyper-parameters are chosen based on the following metric (mt) on the NTU data,

$$mt = \frac{1}{T_p} \sum_{t=T_l.f}^{T_u.f} (acc_t + \frac{1}{N_v} \sum_{i=1}^{N_v} prob_{i,t} - \frac{1}{N_s} \sum_{j=1}^{N_s} std_{j,t}), \quad (4.8)$$

where acc_t denotes overall classification accuracy, $prob_{i,t}$ denotes predicted probability of the appropriate class (e.g., ‘not-crossing’ in stopping sequences) for sequence i and $std_{j,t}$ is standard deviation in probability values among sequences of type j at t . f is the dataset framerate (15 fps for both datasets). The prediction interval is given by $[T_l T_u]$ s and $T_p = (|T_u.f - T_l.f| + 1)$. N_v is the number of considered validation sequences and N_s is the number of considered sequence types (2, viz., ‘continuous crossing’ and ‘stopping’ for NTU data).

4.9.2 Daimler Dataset

Fig. 4.11 presents a comparison of our approach with the best results presented in [7]. ‘0’ along x-axis represent the event (‘crossing’ or ‘stopping’) instants and are labeled for all test sequences in the dataset. As stated earlier, we find vehicle context irrelevant within the Daimler dataset and hence only test our motion based approach for intention prediction, i.e. here $x_{t,Daim} = \{x_{m,t}\}$. We apply a FLDCRF with 1 layer and 4 states per label over our motion features.

The best results in [7] are obtained from a Gaussian Process Dynamic Model (GPDM) applied on dense optical flow derived features from stereo images. As we can see from Fig. 4.11, our motion features with FLDCRF perform comparably to the results in [7]. While our approach performs marginally better for the stopping scenarios, [7] is more consistent through time over the crossing sequences. Our method although enjoys the liberty of monocular images, thus not requiring stereo setup.

Although our primary novelty lies on the influence of vehicle context, our proposed motion features (see Section 4.7.2) are computationally quite inexpensive and still perform comparably to the state-of-the-art [7] performance on the Daimler dataset. In fact, our motion-model marginally outperforms [7] at early prediction (before ‘0’) instances. We now augment the motion model with spatial and vehicle context features on the NTU data.

4.9.3 NTU Dataset

We compare three different systems - mtn (m), $mtn + spa$ (ms) and $mtn + spa + veh$ (msv) with FLDCRF and LSTM on the NTU dataset. ‘0’ along x-axis (see Figures 4.12 - 4.16) represents the event (*crossing*, *stopping* etc.) instants. Positive and negative values along x-axis indicate gain and loss in prediction time (in seconds)

Figure 4.11: Performance comparison between Daimler benchmark [7] and our motion based approach.

respectively. Since we try to predict 2 sec (30 frames) ahead of the event, we will limit our analysis within $[2 - 0.5]$ s prediction interval.

Model hyper-parameters were chosen by considering the value of the metric in (4.8) within $[2 - 0.5]$ s interval. Table 4.3 presents a comparison between LDCRF (FLDCRF with 1 hidden layer), FLDCRF and LSTM over nested validation sets on $mtn + spa + veh$ (msv) systems. Two layered FLDCRF settings (2/1, 2/2, 2/3) performed better than the 1 layered settings (1/1, 1/2, 1/3, 1/4, 1/5, 1/6) within most of the inner validation sets. The new models (2/1, 2/2, 2/3) also help to enhance the overall FLDCRF performance. LSTM models perform worse than the FLDCRF models on the validation sets in general. We select best performing FLDCRF and LSTM settings for each of the three systems on the inner validation sets and average their performance on the outer folds to obtain our results.

Fig. 4.12 displays the performances of different systems on 35 stopping sequences of the NTU dataset. Motion-only FLDCRF model ($FLDCRF - m$) has consistently low average stopping probability (see Fig. 4.12a) and classification accuracy (see Fig. 4.12b) values than models with context in the earlier regions of the curves (before ‘0’). Adding location context to FLDCRF ($FLDCRF - ms$) slightly improves time, probability and accuracy of predicting the stopping intention in earlier regions. The results improve significantly with higher average stopping probability and accuracy on adding vehicle context features to the FLDCRF model ($FLDCRF - msv$). On average, all LSTM models perform worse than the corresponding FLDCRF models in terms of prediction time, accuracy and probability values.

Fig. 4.13 compares different models on stopping scenarios aided by positive vehicle

Table 4.3: Comparison between FLDCRF, LDCRF and LSTM over nested validation sets on $mtn + spa + veh$ system (on the NTU data).

Valid. set \ Model	1	2	3	4	5	Average performance (mt)
FLDCRF-1layer	1.4324	1.4125	1.4447	1.4458	1.4436	1.4358
FLDCRF-2layers	1.4330	1.4199	1.4447	1.4419	1.4439	1.4366
FLDCRF-overall	1.4330	1.4199	1.4447	1.4458	1.4439	1.4374
LSTM	1.3315	1.3355	1.3077	1.4085	1.3706	1.3508

(a) Stopping probability vs time.

(b) Classification accuracy vs time.

Figure 4.12: Performance of different systems on stopping sequences of NTU dataset.

context ($ncl \geq 0$ and/or $ncr \geq 0$). At each instant of the plots, only frames with such vehicle context values are considered. While we find models without vehicle context (mtn and $mtn + spa$) to perform similarly to the earlier results, models with vehicle context ($mtn + spa + veh$) display an enhanced prediction performance with better

Figure 4.13: Performance for stopping sequences with positive vehicle context in the NTU dataset.

average probability and accuracy in earlier regions. An average stopping probability of 0.6 and a classification accuracy of 0.7 was achieved and consistently improved from ~ 0.9 seconds before the actual stopping events by *FLDCRF* – *msv* system. The vehicle context aids a gain of ~ 0.5 seconds in prediction time over *FLDCRF* – *ms* system. *LSTM* – *msv* performance also improves in these scenarios, but the accuracy values still fail to approach 1 (like *FLDCRF* models) in the vicinity of the stopping event.

Different systems are compared in Fig. 4.14 on 35 continuous crossing sequences of the NTU dataset. Accuracy-wise, all *mtn* and *mtn+spa+veh* type models display quite consistent and accurate (close to 1) performance across time. Although, joint models (*mtn+spa+veh*) output smaller average probability values indicating better reliability of the system. *LSTM* and *FLDCRF* joint (*mtn+spa+veh*) models perform comparably across time, with *FLDCRF* type model exhibiting marginally better probability and accuracy values at certain time points.

Figure 4.14: Results on continuous crossing sequences of the NTU dataset.

Figure 4.15: Overall classification accuracy of different systems on the NTU dataset.

Overall accuracy of considered systems at different prediction horizons on NTU dataset is displayed in Fig. 4.15. '0' in the figure represents event (*stopping* and *crossing*) instants of respective sequences. *FLDCRF* – *msv* system proves to be the most accurate at majority of early prediction instants, while performing comparable to certain systems (*FLDCRF* – *ms* and *FLDCRF* – *m*) after the '0' mark, bolstering itself to be selected as the primary choice for an early and accurate intention prediction system on the NTU data.

4.10 Failed Cases and Stability Analysis

Average stopping probability and classification accuracy do not reflect on the system performance for stable and early prediction on individual pedestrian sequences. We present individual sequence predictions by our best ($FLDCRF - msv$) system here. We also analyze the failed predictions by this system.

Fig. 4.16 shows individual sequence outputs by the $FLDCRF - msv$ system. We highlight sequences (in bold red) where the system fails to make early prediction (i.e., before ‘0’) of the events.

All crossing events were predicted correctly before respective *crossing instants* (see Fig. 4.16a). An unwanted spike can be observed on sequence *crossing_17* near the ‘0’ mark, caused by an approaching vehicle. However, the system was able to correct the prediction before the *event instant* to avoid any critical failure.

Individual probability outputs of 35 stopping sequences by $FLDCRF - msv$ system are displayed in Fig. 4.16b. The system fails to make early prediction (before ‘0’) of the stopping event in the two highlighted sequences (by bold red), *stopping_9* and *stopping_32*. Both such cases occur due to the unavailability of vehicle context in the sequences.

Figure 4.16: Individual sequence outputs by $FLDCRF - msv$ on the NTU dataset. Failed cases are shown in bold red.

4.11 Conclusion

We proposed a joint (context + motion) model for pedestrian intention prediction, and introduced the vehicle interaction context for earlier and more accurate prediction. Our system significantly improves the prediction time and accuracy over the

existing approaches and shows promise for safe and smooth autonomous rides on real-life scenarios. The joint *FLDCRF* – *msv* system performs the earliest and most accurate predictions of the stopping behaviour among considered systems and provides better reliability on crossing sequences at the same time, indicating the power of including vehicle context and better performance of FLDCRF over LSTM on the NTU dataset. In general, FLDCRF type models performed much better (with stable prediction probability and better prediction accuracy) than LSTM models across different systems. In the next chapter, we investigate our approach on real-life pedestrian behaviour prediction. The real-life scenarios come with additional challenges, e.g., moving ego-vehicle, pedestrian behaviours other than considered in this chapter, data availability, variable pedestrian density etc. We address to these challenges and propose revised models for applications on two real-life datasets, viz., JAAD [23] and Little India (LI [5]) data in the next chapter.

Chapter 5

Real-life Pedestrian Intention Prediction

We extend our intention prediction models to real-life pedestrian behaviours in this chapter. Tackling real-life pedestrian behaviour brings additional challenges, e.g., moving ego-vehicle, more variations in pedestrian behaviour, variable pedestrian density etc. We discuss these challenges in detail and describe how we address them in our revised approach. We apply the revised models to two real-life datasets - Joint Attention for Autonomous Driving (JAAD [23]) data and Little India (LI [5]) data.

5.1 New Challenges from Real-life Data

Pedestrian behaviours observed from a vehicle varies widely in real-life. Therefore, it is hard to precisely encode the scenarios of interest for AV/ADAS systems. Variable pedestrian density, the moving ego-vehicle additionally create scenarios with variable criticality. We discuss the primary challenges of tackling real-life pedestrian data below.

5.1.1 Moving Ego-vehicle

The moving ego-vehicle brings about the following challenges:

- *Ego-motion in the camera images:* The video data captured from a moving camera (mounted on the ego-vehicle) includes spurious ego-vehicle motion for the objects of interest, e.g., pedestrians, other vehicles etc. An example is illustrated in Fig. 5.1, where moving camera images produce non-zero optical flow values even on the pixels of a stationery pedestrian. Therefore, such unwanted

ego-motion must be compensated from the motion of the objects of interest, in order to avoid misleading predictions.

- *Situation criticality:* The variable ego-vehicle speed together with variable pedestrian density in the real-life data generate situations of various criticality (low to high) measures.
- *Variation in pedestrian behaviour with ego-vehicle speed:* Different ego-vehicle dynamics cause different pedestrian behaviour, based on the ego-vehicle distance and speed from a pedestrian. In other words, the pedestrian and ego-vehicle dynamics interact and influence each other in real time. Such a problem did not exist in the NTU data. Such spontaneous interacting behaviour is also not observed in the Daimler data [37], since the pedestrians and the ego-vehicle were both part of the acting process.

Figure 5.1: Ego-motion compensation from optical flow. (a) Original image [37]. (b) Uncompensated optical flow [32]. (c) Ego-compensated optical flow by [33]. The normalized flow values are plotted in grayscale.

5.1.2 More Variations in Pedestrian Behaviour

Pedestrian behaviour varies widely in real-life. AV/ADAS systems require precise encoding and detection of the scenarios of interest, and need to take decisions accordingly. Therefore, the variable pedestrian behaviour in real-life creates the following challenges for an autonomous system:

- *Encode the behaviours by a precise rule:* All pedestrian behaviours of interest for a particular vehicle can be categorized broadly into two categories: ‘crossing’ before the vehicle, and ‘not-crossing’ before the ego-vehicle. However, in order

to train and evaluate an early prediction machine learning model, we need measures for the early prediction. Therefore, we need to encode all possible crossing and not-crossing scenarios into precise and meaningful categories.

- *Decision making:* The AV/ADAS system not only needs to detect a particular pedestrian behaviour, but also require precise decision making models according to the situation. Therefore, it is preferable to encode important pedestrian behaviours by a low-dimensional alphabet, in order to simplify the decision making process.

In Chapter 4.5, we presented a nomenclature to describe certain pedestrian encounters. However, there can be scenarios (illustrated in Fig. 5.2) where the pedestrian is initially stationery (laterally), and may or may not start crossing before the vehicle. Such cases are not covered by the nomenclature in Fig. 5.2. Therefore, we need a revised nomenclature to include these behaviours.

(a) Continue to be laterally static.

(b) Laterally static then laterally moving.

Figure 5.2: Examples of real-life pedestrian encounters not covered in Chapter 4.

5.1.3 Variable Pedestrian Density

A vehicle navigates through sparse to densely populated areas in real-life. Such variation creates additional challenges for predicting the pedestrian behaviour, as follows:

- *Situation criticality:* The situation criticality varies with the variable pedestrian density, thus influencing the decision making process for designing the vehicle dynamics.
- *Pedestrian pedestrian interactions:* In a crowded scene, pedestrian intentions often tend to vary in a group, i.e., the crossing intention of one pedestrian may

depend on others. Therefore, we need to consider such pedestrian-pedestrian interactions in the model for better prediction of the crossing behaviour.

Figure 5.3: Variable pedestrian density areas while navigating. Figures from LI [5] and JAAD [23] datasets.

5.1.4 Data Availability

Due to privacy reasons and the risks of capturing spontaneous pedestrian-vehicle interactions, there have rarely been pedestrian dedicated datasets with variable pedestrian density and pedestrian behaviours, that could simulate the real-life navigation of autonomous systems.

At the time (in 2016) we started this research, there was no real-life data available for pedestrian behaviour prediction. Recently a group of researchers released JAAD [23] dataset, which is the first released real-life pedestrian dedicated dataset. We evaluate our approach on this data. However, in order to make our system robust, it is necessary to evaluate across multiple real-life datasets. The JAAD data also lacks pedestrian-heavy areas.

In addition, the real-life data also has challenges like occlusion during navigation, night scenarios (where pedestrian detection and semantic segmentation becomes challenging, see Section 5.6.2), rainy days etc. We plan to consider each such challenge with diligence in the near future, in our quest for a fully operational robust pedestrian model for autonomous systems.

5.2 How we respond to the challenges?

We describe here how we respond to the new challenges posed by the real-life pedestrian data.

Figure 5.4: The revised intention prediction system for real-life data.

5.2.1 The revised IP model

In order to account for the changes associated with a moving ego-vehicle, we revise our intention prediction system according to Fig. 5.4. We introduce the ego-vehicle velocity and combine it with the ego-vehicle distance to obtain the ego-vehicle context for each considered pedestrian. We also include a ego-compensation system [33] to compensate for the ego-motion from computed optical flow values (see Fig. 5.1b-c). Our pedestrian motion feature extraction system is robust to the scenarios of a moving camera (see Chapter 4.7.2), as the process involves compensating the entire longitudinal motion, which includes the motion of the moving vehicle.

5.2.2 The Revised Nomenclature

Pedestrian behaviour varies widely in real-life. However, all pedestrian behaviours of interest for a particular vehicle can be categorized broadly into two categories: ‘crossing’ before the vehicle, and ‘not-crossing’ before the ego-vehicle. In order to train and evaluate an early prediction machine learning model, we need measures for the early prediction. Therefore, we need to encode all possible ‘crossing’ and ‘not-crossing’ scenarios into precise and meaningful categories, possibly based on a single key insight.

Key insight:

A crossing pedestrian in front of the ego-vehicle always have a lateral motion component w.r.t. the ego-vehicle moving direction.

Therefore, in order to accommodate majority of crossing/non-crossing pedestrian

encounters from a vehicle, we propose a revised nomenclature (see Fig. 5.5) on pedestrian interactions from a moving vehicle in real world. We introduce two new behaviour types (compared to Chapter 4.5), which include the cases of an initially laterally static pedestrian, who may remain laterally static or start crossing in future in front of the vehicle.

We evaluate on sequences where early prediction is relevant from an AV perspective. We categorize each such encountered pedestrian sequence into one of the following (see Fig. 5.5)¹:

1. **Continuous crossing:** The pedestrian initially has a lateral movement before the curb, continues to move laterally and crosses in front of the vehicle.
2. **Stopping at the curb:** The pedestrian initially has a lateral movement before the curb and stops at/near the curb. We call this behaviour ‘stopping’ in this chapter for simplicity.
3. **Standing at the curb:** The pedestrian continues to be laterally static at/near the curb. This behaviour is referred as ‘standing’ in the chapter.
4. **Starting to cross:** The pedestrian is initially static laterally at/near the curb and starts to cross in front of the vehicle. This behaviour is called ‘starting’ in the chapter.

Stopping and starting sequences carry a change in state of motion (laterally moving to laterally static and vice versa) of the pedestrian. As our goal is to predict pedestrian intention early, our primary aim is to detect these state changes early (preferably before any discernible change in motion/appearance) and accurately. In addition, we also want stable and accurate predictions of ‘crossing’ and ‘not-crossing’ intentions for sequences with no change in state of motion (continuous crossing and standing respectively).

Each sequence type is associated with an *event instant* (e.g., *crossing*, *stopping* or *starting instant*), illustrated in Fig. 5.5, which provides the ground truth classification labels and also serves as reference during evaluation. Since a standing sequence cannot be characterized with such an instant, we define a *critical point* based on pedestrian distance and ego-vehicle velocity. To aid the labeling process during training (for early prediction of the transitions in stopping and starting sequences), we define a *pred_ahead* parameter for each dataset, in number of frames before the

¹Associated critical measures and necessary vehicle dynamics can be derived by further analyzing pedestrian’s position w.r.t. the vehicle.

Figure 5.5: Examples of common pedestrian sequence types observed from a vehicle (from JAAD [23] dataset). EI stands for event instant (crossing, stopping etc.).

respective event instants. We explain how we labeled the pedestrian sequences during training in Section 4.8.

5.2.3 A New Dataset: Little India (LI) Data

To achieve robustness across multiple datasets, we collected our own data (LI dataset²) in downtown Singapore, covering low to highly dense pedestrian areas (e.g., Little India, Geylang). We evaluate our models on both JAAD and LI datasets.

5.2.4 Pedestrian Interaction Model

To include pedestrian-pedestrian interactions for densely populated areas, we are revising our model with the interaction variant of FLDCRF (FLDCRF-i, see Fig. 5.6, see Chapter 3 for details). However, we are still in the process of applying this interaction model on the LI data, and hopefully soon publish our results.

²by a joint effort with ST Engineering

Figure 5.6: FLDCRF graphical model for pedestrian-pedestrian interactions. Two different pedestrians, $(\{x_{1,t}\}, \{y_{1,t}\})$ and $(\{x_{2,t}\}, \{y_{2,t}\})$ are shown to be interacting in the latent space through their hidden layers to effect each others' dynamics. Longer range dependencies are possible but are omitted for to avoid clutter. The vehicle context is included in the input features $x_{i,t}$ of pedestrian i . A spatial grid (b), similar to [21], is included to allow only nearby pedestrians to interact.

5.3 Datasets

We briefly describe characteristics of the considered real-life JAAD [23] and LI [5] datasets below.

5.3.1 JAAD

The JAAD dataset [23] is a recent pedestrian benchmark which captures real-world pedestrian behaviours in complex scenes from cameras mounted on moving vehicles. It contains 346 video clips at 1920×1080 resolution (few at 1280×720) and 15 fps framerate, each containing multiple pedestrian sequences. All pedestrians of interest are annotated by bounding boxes and are assigned certain movement labels (moving, crossing, stopping, standing, etc.) and personal details (age, sex, movement direction, etc.). Each frame in the videos is also associated with certain traffic attributes, such as zebra crossing, parking lot, etc.

To our best knowledge, there is no JAAD data subset proposed for early pedestrian intention prediction. We extract a total of 120 pedestrian sequences distributed across continuous crossing, stopping, starting, and standing scenarios (see Table 5.1). The sequence details are provided below.

JAAD Sequence details

Considered JAAD sequences are presented in Fig. 5.7. A pedestrian sequence is denoted by its *video_id* and *pedestrian_id* in JAAD dataset, in the following format: $\langle video_id \rangle_ \langle pedestrian_id \rangle$.

Direct crossing		Stopping	Starting		Standing
0002_1	0215_2	0037_1	0057_1	0242_3	0007_1
0002_2	0224_3	0059_1	0070_1	0244_2	0008_1
0006_1	0224_4	0089_1	0072_1	0244_3	0068_1
0006_2	0224_5	0090_1	0072_2	0248_1	0071_1
0024_1	0231_1	0107_1	0080_1	0249_1	0195_1
0027_1	0231_3	0128_1	0080_2	0275_1	0208_2
0044_1	0231_4	0147_1	0090_1	0277_1	0209_3
0049_1	0259_1	0183_1	0140_1	0277_3	0211_1
0049_2	0260_1	0209_4	0165_2	0286_1	0239_1
0061_1	0276_1	0237_2	0173_1	0287_1	0245_1
0063_1	0290_1	0241_1	0176_1	0299_1	0254_1
0071_2	0290_2	0249_1	0183_1	0299_2	0264_2
0079_1	0297_1	0277_1	0190_1	0301_1	0314_2
0088_1	0297_2	0277_3	0196_1	0302_2	0314_4
0120_1	0297_3	0286_1	0228_2	0302_3	0325_2
0131_2	0297_4	0294_1	0231_2	0302_4	
0132_1	0303_1	0295_1	0235_1	0306_1	
0132_2	0305_8	0302_2	0235_2	0310_2	
0147_2	0313_2	0302_3	0236_2	0315_1	
0161_1	0313_3	0302_4	0241_1	0326_1	
0166_1	0320_2	0334_2	0242_1		
0177_2	0320_3	0336_1	0242_2		
0183_2					

Figure 5.7: Considered JAAD pedestrian sequences in our evaluation.

5.3.2 Little India data

We collected the Little India (LI [5]) data in downtown Singapore, focusing on crowded areas like Little India and Geylang. The data is captured by a GoPro 4K camera mounted on top of a vehicle. The videos have a 1920×1080 resolution and 15 fps frame rate. We collected 44 clips ranging from 30 seconds to 3 mins in length.

We processed the raw data to remove segments without pedestrians. The processed data has 201 clips. We ran the Faster R-CNN [34] to detect and track pedestrians and vehicles in the dataset. There are over 5000 pedestrian tracks, ranging from very sparse (1 pedestrian) to very dense areas (>100 pedestrians). We are still processing this data to prepare the pedestrian sequences, e.g., labeling, tracking continuity, association with the nomenclature etc. For evaluation in this thesis, we processed 50 pedestrian sequences distributed across continuous crossing, stopping, starting, and standing scenarios (see Table 5.1).

5.4 Feature Extraction

The location context $x_{sp,t}$ (or spatial context) and pedestrian motion $x_{m,t}$ feature extraction process for applications on the real-life datasets in this chapter are same as Chapter 4.7. We modify the vehicle context features to include the ego-vehicle context as follows.

5.4.1 Ego-vehicle Context

The ego-vehicle speed and distance are important factors to determine the crossing intention of pedestrians in front of the vehicle.

We utilize the provided ego-vehicle velocity labels in the JAAD dataset. At each instant, the ego-vehicle is assigned one of these velocity labels: ‘moving slow’, ‘moving fast’ or ‘standing’. Some instants only have the ego-acceleration labels (‘slowing down’ and ‘speeding up’) and do not contain velocity labels. We associate velocity labels to these instants by imputation/observation. We ignore the ego-acceleration labels in this thesis. We followed a similar ego-velocity labeling for the LI data.

Similar to Chapter 4.7.1, we build the ego-vehicle depth model by pre-calibrating the scene. We obtain depth (longitudinal distance) of the pedestrians by applying this model to the bottom center pixel of the pedestrian bounding boxes.

The complete vehicle context for the JAAD and LI datasets is given by, $x_{v,t} = \{ego_dep, ego_vel\}$, where *ego_dep* is the estimated pedestrian depth from the ego-vehicle and *ego_vel* is the ego-velocity attribute (‘moving slow’, ‘moving fast’ etc.). As the other vehicles do not influence pedestrian behaviours in most cases within JAAD and LI dataset, we keep things simpler by only considering the ego-vehicle context. We defer more complete interactions including other vehicles in the scene to our future work, as we need more data to train such instances.

Therefore, the complete feature set for the models in this chapter is given by, $x_t = \{x_{m,t}, x_{sp,t}, x_{v,t}\}$, where $x_{m,t}$ and $x_{sp,t}$ are obtained from the methods described in Chapter 4.7, and the $x_{v,t}$ are computed by the process described above.

5.5 Experimental Setup

We describe here our experimental data, extracted features and labels and the model specifications.

5.5.1 Data for evaluation

As mentioned earlier, there was no JAAD data subset proposed for early pedestrian intention prediction prior to our work. We extracted a total of 120 pedestrian sequences distributed across continuous crossing, stopping, starting, and standing scenarios (see Table 5.1). We also extract 50 pedestrian sequences in the four categories (see Table 5.1) for our evaluation in the Little India data.

We apply Faster R-CNN [34] to detect and track pedestrians and vehicles in the Little India data. The object detector could efficiently detect pedestrians and vehicles up to 80 meters (at ~ 0.02 sec/frame with downsampled frame size 960×540) from the camera. For pedestrian sequences in the JAAD data, we utilize provided bounding box annotations.

The distribution of test sequences across the two datasets is presented in Table 5.1.

Table 5.1: Table showing distribution of test sequences in different datasets.

Seq type Dataset	Continuous crossing	Stopping	Starting	Standing	Total # instances
JAAD	45	20	40	15	19650
LI ³	20	10	5	15	3395

5.5.2 Features and Labels

The considered complete feature set on the JAAD and the LI data is given by, $x_t = \{x_{m,t}, x_{sp,t}, x_{v,t}\}$.

For both JAAD and LI dataset, we set $pred_ahead = 20$ (1.33 sec). Due to additional types of pedestrian behaviours (standing and starting, see Section 5.2.2), we propose a revised labeling mechanism (compared to Chapter 4, see Fig. 4.10) of the pedestrian sequences during training. The revised labeling is illustrated in Fig. 5.8.

³We are still processing this data to extract more pedestrian sequences for evaluation. We will have more pedestrian sequences on this dataset in future.

Figure 5.8: Labeling the training sequences in the JAAD and LI data. ‘c’ denotes ‘crossing’ and ‘n’ denotes ‘not-crossing’ intention.

5.5.3 Models compared

We apply FLDCRF-s (see Chapter 3) and LSTM as our learning models. We show results for three different systems:

1) Only pedestrian motion based, referred as *mtn* (or *m*) in the Results section. This system only utilizes pedestrian motion (see Section 4.7.2) as input features, i.e., $x_t = \{x_{m,t}\}$.

2) Pedestrian motion and location based, referred as *mtn + spa* (or *ms*). This system takes pedestrian’s semantic location (see Section 4.7.1) along with pedestrian motion as inputs, i.e., $x_t = \{x_{m,t}, x_{sp,t}\}$.

3) Pedestrian motion and vehicle context based, referred as *mtn + veh* (or *mv*). This system considers vehicle context features (see Section 5.4.1) together with pedestrian motion, i.e., $x_t = \{x_{m,t}, x_{v,t}\}$. For both JAAD and LI data, the semantic segmentation [35] performance is unsatisfactory, resulting in poorer performance (see Section 5.6.2) on including pedestrian location information. Therefore, we omit pedestrian location information from this final model, which takes pedestrian motion and vehicle context features.

We employ nested cross-validation (with 5 outer folds and 4 inner folds) for selecting models and generating results. For selecting LSTM models, we tested the following number of hidden units (*HU*): 2, 5, 10, 20, 50, 150, 300, 500. We trained each model for 1000 epochs, saving model performances at every 100 epochs during validation. We fed all training sequences at each epoch at the rate of 1 sequence per batch.

An FLDCRF-s has two hyperparameters: number of hidden layers N_h and number

of hidden states per label $\{N_{si}\}$ along layers $i = 1 : L$. For simplicity, we keep same N_s along all layers, i.e. $N_{si} = N_s, \forall i = 1 : L$. We denote such model by FLDCRF- N_h/N_s . We selected our models from the following N_h/N_s combinations: 1/1, 1/2, 1/3, 1/4, 1/5, 1/6, 2/1, 2/2, 2/3.

We trained the FLDCRF model parameters with Stan's [31] in-built BFGS optimizer. We implemented the LSTM models defined in Keras, a deep learning library for Python running Tensorflow in the backend. LSTM parameters were trained by a default Adam optimizer in Keras. We plotted the results in MATLAB 2015b [36]. LSTM models are trained with a Nvidia Tesla K80 GPU, while the FLDCRF models are trained with a Intel Xeon E5-1630 v3 CPU.

5.6 Results

We compare different systems on the JAAD and LI datasets using existing state-of-the-art (LSTM) and proposed (FLDCRF) sequential models.

5.6.1 Metric definition

We evaluate the models by two metrics: stopping probability vs time and classification accuracy vs time. Stopping probability corresponds to the probability of the 'not-crossing' (laterally static) label. We define a metric for selecting models on the validation data, similar to Chapter 4, as follows.

We ideally want high classification accuracy of appropriate class labels (e.g., 'not-crossing' in stopping sequences) over the prediction interval. At the same time, it is desired to have high and stable average probability values of appropriate class labels over the prediction interval. It is also preferred that the sequences of same type have minor standard deviation in predicted probability values at each instant. So, our model hyper-parameters are chosen based on the following metric (mt),

$$mt = \frac{1}{T_p} \sum_{t=T_l.f}^{T_u.f} (acc_t + \frac{1}{N_v} \sum_{i=1}^{N_v} prob_{i,t} - \frac{1}{N_s} \sum_{j=1}^{N_s} std_{j,t}), \quad (5.1)$$

where acc_t denotes overall classification accuracy, $prob_{i,t}$ denotes predicted probability of the appropriate class (e.g., 'not-crossing' in stopping sequences) for sequence i and $std_{j,t}$ is standard deviation in probability values among sequences of type j at t . f is the dataset framerate (15 fps for both datasets). The prediction interval is given by $[T_l T_u]$ s and $T_p = (|T_u.f - T_l.f| + 1)$. N_v is the number of considered validation sequences and N_s is the number of considered sequence types (4 for both JAAD and LI data).

5.6.2 JAAD

On the JAAD dataset, we evaluate on all four types of sequences: 45 continuous crossing, 20 stopping, 40 standing and 15 starting (see Section 5.2.2).

We compare results from FLDCRF and LSTM on three different systems: mtn (m), $mtn + spa$ (ms) and $mtn + veh$ (mv). Since the JAAD data contains variety of scenarios (some during night-time), semantic segmentation [35] performance is not consistent. This results in $mtn + spa$ models performing slightly worse than only mtn models (see Fig. 5.9). Therefore, we omit pedestrian location information from our final model, which takes pedestrian motion ($x_{m,t}$) and vehicle context ($x_{v,t}$) features.

We obtain our test results on JAAD data from a 5-fold nested CV. We make use of the metric in (5.1), within [1.33 -1] s interval, to select hyper-parameter settings on inner validation sets. FLDCRF (1.3640) improves over the LDCRF (1.3627) average performance on the validation sets.

Fig. 5.9 presents results of the three different systems, m , ms and mv , trained with FLDCRF and LSTM. It can be clearly seen that the models with vehicle context, $FLDCRF - mv$ and $LSTM - mv$ perform significantly better than models without the vehicle context across all behaviour types. Although $LSTM - mv$ performs marginally better than $FLDCRF - mv$ at certain instants on stopping and standing sequences, $FLDCRF - mv$ consistently outperforms $LSTM - mv$ on the continuous crossing and starting sequences. Furthermore, $LSTM - mv$ lacks stability in accuracy across time on the critical continuous crossing sequences, whereas $FLDCRF - mv$ produces stable, early and accurate prediction of the crossing behaviour, with more than 95% average accuracy in the important region of [1 -0.5] s.

We compare different systems on the task of predicting early transitions (standing-starting and crossing-stopping) in Tables 5.2 and 5.3. We consider accuracy of the systems over stipulated prediction windows. As expected, systems with vehicle context $FLDCRF - mv$ and $LSTM - mv$ perform significantly better than systems without vehicle context. $FLDCRF - mv$ outperforms $LSTM - mv$ in all early prediction windows. However, $LSTM - mv$ performs marginally better than $FLDCRF - mv$ after the '0' mark on crossing and stopping scenarios considered together (see Table 5.3).

$FLDCRF - mv$ and $LSTM - mv$ consistently performed well across all sequence types. Other systems have been better at times but lacked consistency. We can observe dominant performance of systems with vehicle context ($FLDCRF - mv$ and $LSTM - mv$) in Fig. 5.10, producing superior overall accuracy compared to systems without vehicle context at all prediction instants. Moreover, $FLDCRF - mv$ performs similar/better than $LSTM - mv$ at all considered prediction instants,

(a) Continuous crossing sequences.

(b) Continuous crossing Sequences.

(c) Stopping sequences.

(d) Stopping Sequences.

(e) Starting sequences.

(f) Starting Sequences.

(g) Standing sequences.

(h) Standing Sequences.

Figure 5.9: Accuracy performances of different systems vs time on the JAAD data.

Table 5.2: Classification accuracy of all considered JAAD standing & starting sequences by different systems within different prediction windows.

System	Classification Accuracy(%)					
	2-0 s	1.5-0 s	1-0 s	0.5-0 s	0-(-0.5) s	0-(-1) s
FLDCRF- (mtn+veh)	56.02	57.98	61.02	68.18	77.17	82.50
FLDCRF- (mtn+spa)	45.84	48.17	50.11	54.32	72.53	78.52
FLDCRF- (mtn)	46.15	47.84	50.45	55.23	70.91	78.18
LSTM- (mtn+veh)	51.82	53.15	56.48	60.91	71.92	77.95
LSTM- (mtn+spa)	35.24	36.84	38.07	40.91	56.16	66.82
LSTM- (mtn)	33.59	34.46	37.05	42.05	54.55	66.36

Table 5.3: Classification accuracy of all considered JAAD continuous crossing & stopping sequences by different systems within different prediction windows.

System	Classification Accuracy(%)					
	2-0 s	1.5-0 s	1-0 s	0.5-0 s	0-(-0.5) s	0-(-1) s
FLDCRF- (mtn+veh)	90.47	91.39	91.83	93.08	93.68	95.29
FLDCRF- (mtn+spa)	74.85	75.57	77.21	78.85	78.63	79.62
FLDCRF- (mtn)	75.93	76.31	77.88	80.19	78.29	80.67
LSTM- (mtn+veh)	86.73	87.94	90.10	93.08	95.90	95.38
LSTM- (mtn+spa)	73.53	75.77	77.12	80.38	82.39	85.87
LSTM- (mtn)	75.87	76.79	77.4	78.46	83.93	87.5

proving to be the best performing model on the JAAD dataset. *FLDCRF* – *mv* produces an average accuracy of $\sim 78\%$ within the 1-0 s window and $\sim 82\%$ within the 0.5-0 s window, while *LSTM* – *mv* outputs $\sim 75\%$ and $\sim 78\%$ in respective windows.

JAAD dataset is not equipped with precise camera matrix information for the video sequences. Consequently, we utilized approximate values for these parameters, and observe our motion features to perform weaker than NTU dataset (see Chapter 4). However, combined with context features we obtain relatively early and accurate intention prediction for all four types of sequences from the *FLDCRF* – *mv* and *LSTM* – *mv* systems. Next, we investigate our approach on the LI data, where we have the precise camera matrix information. Consequently, we will visualize the motion features performing substantially better than the JAAD data.

Figure 5.10: Overall classification accuracy of different systems on JAAD sequences.

5.6.3 Little India data

We present results for different FLDCRF and LSTM models on the LI data here.

Fig. 5.11 presents results of different systems on the LI data. We do not show the *ms* (motion and location) systems as they performed worse than the motion-only (*m*) systems. It should also be noted that, we simply applied cross-validation with a FLDCRF-s(2/2) and LSTM with $N_{hls} = 2$ and $N_{els} = 500$ to obtain our results. Due to time limitations, we could not apply nested cross-validation on the LI data, rather selected good performing FLDCRF and LSTM hyperparameters from earlier experiments. The performances can be further improved by optimization, especially for LSTM, which almost always requires tedious hyperparameter optimization, as shown in several experiments in this thesis.

Similar to our results on the JAAD data, *FLDCRF* – *mv* system outperforms all other systems, with significantly better stopping probability and classification accuracy outcomes across sequence types. The overall dominance of this system is further reflected in Fig. 5.12, where we plot the overall classification accuracy across sequence types. The inconsistent performance by *LSTM* – *mv* with random (but verified from earlier experiments) hyperparameters serves as a further reminder of the difficulties associated with optimizing the LSTM models and their consistency on the test data. On the other hand, a randomly chosen FLDCRF setting performs reasonably well across sequences.

The motion-only models perform significantly better on the LI data than the JAAD data, due to precise camera matrix information. Consequently, the joint model *FLDCRF* – *mv* also performs notably better than on the JAAD data, due to the strong foundation laid by good motion features.

(a) Continuous crossing sequences.

(b) Continuous crossing Sequences.

(c) Stopping sequences.

(d) Stopping Sequences.

(e) Starting sequences.

(f) Starting Sequences.

(g) Standing sequences.

(h) Standing Sequences.

Figure 5.11: Comparison of different systems vs time on the LI data.

Figure 5.12: Overall classification accuracy of different systems on LI sequences.

5.7 Failed Cases and Stability Analysis

Average stopping probability and classification accuracy do not reflect on the system performance for stable and early prediction on individual pedestrian sequences. We present individual sequence predictions by our best (*FLDCRF – mv*) system here. We also analyze the failed predictions by this system.

5.7.1 JAAD data

We analyze individual sequence outputs and failed cases by the *FLDCRF – mv* system on the JAAD data below.

Fig. 5.13 depicts failures in bold red from *FLDCRF – mv* system. We denote a pedestrian sequence by its *video_id* and *pedestrian_id* in JAAD dataset, i.e., $\langle video_id \rangle_ \langle pedestrian_id \rangle$.

Crossing sequences 0161.1 and 0177.2, where the system fails to establish stable and accurate (stopping probability < 0.5) outputs after -0.25 s, are highlighted in Fig. 5.13a. In sequence 0161.1, we find a momentary prediction glitch within [-0.5 -1] s window caused by a temporary hesitance from the pedestrian to continue crossing. The vehicle happened to be quite far from the pedestrian during the glitch avoiding a critical failure by the system. The failure in sequence 0177.2 is caused by inappropriate vehicle context as the ego-vehicle is moving fast quite near (< 15 m) the pedestrian when the crossing event commenced in a different lane. However, the vehicle decelerated within a short period and stable, accurate output was achieved shortly after the -1.5 s mark. Such errors can be corrected by adding the lane information as context.

All stopping scenarios were predicted correctly by the *FLDCRF – mv* system, latest by 0.75 s after the stopping instant. The system fails to make early prediction (before ‘0’) of the event on the highlighted sequences (0334.2 and 0336.1). A few

Figure 5.13: Individual sequence outputs by $FLDCRF - (mtn + veh)$ on the JAAD dataset. Failed cases are shown in bold red.

sequences fall below probability 0.5 after the -1 s mark, but all such cases correspond to a starting event followed by the stopping event.

We have found early prediction of the ‘starting’ event quite challenging. Fig. 5.13c highlights three starting scenarios that fail to be stable and accurate by -1.5 s on the curves. However, all of them become stable and accurate shortly after the -2 s mark.

Standing sequence 0208_2 is wrongly predicted as ‘crossing’ by *FLDCRF* – *mv* before and well after (≈ 1 s) the defined critical point. A possible reason behind this is incomplete and inaccurate vehicle context. In this case, the other vehicles before the ego-vehicle are primarily responsible for the pedestrian to remain stationery (see Fig. 5.14). The ego-vehicle is moving slowly during the event in a busy scene and is quite far from the pedestrian, causing the inaccurate context. We defined the critical point randomly due to limited number of images in the sequence. In such cases, we need to consider more complete pedestrian-vehicle interactions, which include other vehicles in the scene.

Figure 5.14: Failed prediction by *FLDCRF* – *mv* on standing sequence 0208_02. *cp* deontes the critical point (‘0’ mark). GT is the ground-truth intention at ‘0’.

5.7.2 LI data

As can be seen from Figures 5.11 and 5.12, the *FLDCRF* – *mv* achieves 100% accuracy across the standing and stopping scenarios well before the event instants. It only fails in one crossing sequence before the event instant, which soon is corrected after ‘0’. However, it takes about ‘+1 s’ after the event instant for all starting scenarios to be predicted correctly. We will perform a more detailed analysis of failed cases on this data with a larger dataset, i.e., with more evaluation sequences.

5.8 Discussion

We discuss here the time requirement by our *FLDCRF* – *mv* system and visualize a few prediction outcomes by the best *FLDCRF* – *mv* system.

5.8.1 A Real-time System

A real-time system requires an inference time of <0.067 s, for data with 15 fps frame rate, as in most of the datasets considered in this thesis for pedestrian intention prediction. Our *FLDCRF*–*mv* system (with $N_h = 2$, $N_s = 2$) for real-life pedestrian intention prediction runs in real time, with a total inference time of 0.035 seconds on average after observing a frame. This time includes pedestrian detection (~ 0.02 s per frame on downsampled image size 960×540), feature extraction and model inference. We do not have the optical flow features in our *FLDCRF* – *mv* system, which aids to this possibility of real-time system. In a revised model with other vehicles in the scene, we need to compute the ego-compensated optical flow inside vehicle BBs, as a measure of vehicle velocity, similar to Chapter 4. We need GPU implementations of optical flow computation to make such systems real-time.

5.8.2 Visualization of Prediction Results

We started this thesis by highlighting our objective with four scenarios picked from the JAAD dataset. We display the prediction outcomes by the *FLDCRF* – *mv* system on these four scenarios in Fig. 5.15.

5.9 Conclusion

We extended our context model to real life pedestrian behaviour in this chapter. The models with dynamic vehicle contexts continued to dominate models without such context, across two real-life datasets. *FLDCRF* models also continued to outperform LSTM models across both data. We observed the motion-only model to perform significantly better on the Little India data due to the precise camera matrix. We are working on models to include pedestrian-pedestrian interactions in crowded scenes, for improved prediction. We also are processing the LI data, in order to increase the number of pedestrian sequences for training and evaluation. We will also look to accommodate other vehicles in the model with larger training data. We describe several future research directions on the pedestrian behaviour prediction model in Chapter 6.

(a) Standing - Not crossing.

(b) Stopping - Not crossing.

(c) Stopping - Not crossing.

(d) Continuous crossing.

Figure 5.15: Sample prediction outcomes by the $FLDCRF - mv$ system on the JAAD dataset [23]. A green bounding box indicates ‘not-crossing’, a pink bounding box indicates ‘crossing’ and a red bounding box indicates ‘crossing within 20 meters’.

Chapter 6

Conclusions and Future work

We started this thesis with two existing real-world challenges at hand: to design a early bahviour prediction model for on-road pedestrians and to devise a sequential machine learning model with stable, better and consistent performance. During chapters 3-5 in this thesis, we have proposed models to solve these problems. We conclude in this chapter by summarizing our primary contributions. We also present ideas and precise directions for future research rooted on our work in this thesis.

6.1 Summary of Main Contributions

In the Introduction, we discussed how current Autonomous Vehicles (AV) and Advanced Driver Assistance Systems (ADAS) face a trade-off between on-road pedestrian safety and smoothness of the ride. During the thesis, our primary task has been to design a stable and eary real-life pedestrian behaviour prediction model for future safe and smooth autonomous rides. Since pedestrian-vehicle interactions are continuous in time, we modeled our task as a time-series (behaviour) classification or a sequence labeling problem.

Existing state-of-the-art sequence labeling models, viz., Long Short-Term Memory (LSTM) networks and Latent-Dynamic Conditional Random Fields (LDCRF), either offer better performance (LSTM) or a stable and consistent (LDCRF) performance. We observed such performance during Chapter 3 of this thesis. The (on average) better performing LSTM models are also well known to for big-data-driven performance and tedious model selection process (see Chapter 3.7), thereby adding further constraints for general applications. At the same time, for a critical problem like on-road pedestrian behaviour prediction, a model must offer stability and consistency in prediction along with superior prediction performance.

We proposed Factored LDCRF in Chapter 3 as an alternative solution to se-

quence labeling. FLDCRF not only offered better test performance than existing state-of-the-art models (LSTM and LDCRF) across experiments in this thesis, but also achieved such performance with stability and consistency. The key idea behind FLDCRF is modeling multiple interacting latent-dynamics within the class labels, which aids to deeper (compared to LDCRF) understanding between the input features x_t and the output classification labels y_t , $t = 1 : T$, and thus improved labeling performance. This key idea of interacting latent-dynamics has led to two different sequence models, viz., a multi-label FLDCRF (FLDCRF-m) and a interaction FLDCRF (FLDCRF-i). We show a FLDCRF-m to generalize a Dynamic CRF (DCRF) with embedded hidden variables and outperform DCRF and all other state-of-the-art sequence models (LDCRF, LSTM, LSTM-CRF, and a multi-label LSTM) in a multi-label sequence tagging experiment (see Chapter 3.7). The interaction variant of FLDCRF (FLDCRF-i) allows social agents to influence each others' dynamic behaviour. We intend to examine FLDCRF-i on social interaction modeling tasks, e.g., pedestrian-pedestrian interaction, in multi-robot social environment for task modeling etc. Our key contributions towards a stable, better and consistent sequence labeling model are as follows:

- We proposed a generalization of LDCRF, called Factored LDCRF (FLDCRF), to capture sequential interactions in latent space. FLDCRF offers three unique variants, viz., FLDCRF-s for single-label sequence modeling, FLDCRF-m for multi-label sequence modeling, and FLDCRF-i for modeling interactions in a social environment. We mathematically show FLDCRF to subsume major sequential CRF variants, viz., LDCRF, DCRF and LCCRF.
- Single label variant of FLDCRF (FLDCRF-s) allows multiple latent dynamics of the class labels to interact with each other, leading to deeper understanding. FLDCRF-s outperforms CRF, LDCRF, LSTM and LSTM-CRF models in the single-label sequence tagging experiments.
- Multi-label FLDCRF (FLDCRF-m) outperforms Factorial CRF, coupled CRF, multi-label LSTM models and all considered single label sequence models (CRF, LDCRF, LSTM and LSTM-CRF) on the joint sequence tagging task. We show that while factorial CRF and coupled CRF do not yield much improvement over a CRF, the interaction among the latent dynamics in FLDCRF-m results in substantial improvements (upto 3%) over the individual LDCRF models.
- A model should not only be judged by its one-time performance on the validation/test data, rather there are several modeling aspects one must address to

for selecting a good model. We pen attributes of a good model in Chapter 3.7.1. We compare FLDCRF and LSTM models on all such modeling aspects, viz., test performance, ease of model selection on the validation data, consistency across validation and test performance, and computation times for training and inference (see Section 3.7).

We propose a context model for stable and early pedestrian behaviour prediction in Chapter 4 and extend the model for real-life predictions in Chapter 5. We introduce a key factor for early prediction, i.e., dynamic vehicle interactions on pedestrian crossing/not-crossing intention. We summarize our main contributions towards pedestrian behaviour prediction below:

- We introduce the influence of vehicle interactions (dynamic context) on pedestrian intention for early and accurate prediction. We show such vehicle context to significantly improve prediction time and accuracy across several datasets, viz., NTU [4], JAAD [23] and LI [5], in Chapters 4 and 5.
- We contribute two datasets for research on pedestrian behaviour prediction, viz., NTU data [4] and LI data [5]. NTU data offers simple but spontaneous pedestrian-vehicle interactions for testing intention prediction models. The real-life Little India (LI [5]) offers variety of (very simple to extremely complex) pedestrian-vehicle interactions ranging from low to very high density pedestrian areas.
- Pedestrian behaviour varies widely in real-life. We propose a generic nomenclature on on-road pedestrian behaviours (see Chapter 5.2.2), in order to encode majority of pedestrian-vehicle interactions of interest. The nomenclature utilizes a fundamental concept that, a pedestrian's crossing behaviour in front of a vehicle is characterized by the lateral (w.r.t. the vehicle moving direction) component of the pedestrian motion. We also present a simple formulation of the intention prediction problem.
- Our models simulate real-life prediction scenarios, i.e., includes state-of-the-art object detection [34] and semantic segmentation [35] systems to account for system errors in our inputs.
- The real-life data introduces several other challenges, e.g., moving ego-vehicle, variable pedestrian density and variety of pedestrian interactions. We address to these new challenges and present sophisticated models to account for all such new challenges.

6.2 Future Research on FLDCRF

FLDCRF offers several future research directions, e.g., application on big data, applying the interaction FLDCRF (FLDCRF-i) in social prediction tasks, end-to-end models such as CNN-FLDCRF etc. We describe each possible future research direction on FLDCRF below.

6.2.1 Standard and Larger sequence datasets

As described in Chapter 3, most studies on sequence labeling have been conducted in the context of Natural Language Processing (NLP), viz., CoNLL 2000 chunking [71], CoNLL 2003 named entity tagging [72]. However, recent reported (F1-measure) performance results on these tasks are largely driven by the quality of the input features, while the models do not show much variance in performance. By contrast, we wanted to determine the effectiveness of the proposed FLDCRF models and therefore did not consider such datasets for evaluating our models in Chapter 3. We plan to apply FLDCRF on such standard NLP tasks in the future.

Inspired by FLDCRF performance on similar sized datasets ($\mathcal{O}(10,000)$ instances), we proceed to apply FLDCRF on a relative new research problem of pedestrian intention prediction on the problem-specific datasets. Deep learning with neural networks for sequence modeling (e.g., Recurrent Neural Networks (RNN), LSTM etc.) has particularly been shown to perform better on big data, e.g., CNN-LSTM models [29] for action recognition on UCF-101 [94] and HMDB-51 [95] datasets, a BILSTM-CRF model [53] for CoNLL 2000 chunking [71] task etc. The training data in each of these tasks involve $\geq 100,000$ instances, as compared to datasets employed in this thesis (with $\leq 40,000$ instances).

By varying the number of hidden layers N_h and number of hidden states associated to class labels in each layer N_{si} , $i = 1 : L$ (see Chapter 3), it is possible to fit FLDCRF efficiently to datasets with variable sizes. However, as described earlier, such extensions with higher layers and states increase FLDCRF model complexity and hence a GPU implementation is possibly needed. We look forward to GPU implementations of FLDCRF and intend to apply the FLDCRF models on larger sequence datasets and compare with state-of-the-art RNN and LSTM models.

6.2.2 FLDCRF-i

Fig. 6.1 shows the interaction variant of FLDCRF-i. Due to time limitations and application requirements, we could not apply FLDCRF-i on social labeling tasks

during the thesis period. However, we have solid future plans on applying this model to atleast two different tasks, viz.,

- to model pedestrian-pedestrian interaction in crowded scenes, in order to capture additional dynamic context for better prediction. We describe this application in Section 6.3.1.
- to model multiple interacting mobile robots [96] for tasks like last mile delivery, warehousing, cleaning, inspection, safety and security. As multiple robots may need to co-operate for optimum performance, their tasks are correlated. Such task is popularly known as fleet management [96]. We describe heterogeneous state-space models to accomplish such interactive tasks in Section 6.2.5.

Figure 6.1: FLDCRF graphical model for social interaction. Two different objects, $(\{x_{1,t}\}, \{y_{1,t}\})$ and $(\{x_{2,t}\}, \{y_{2,t}\})$ are shown to be interacting in the latent space through their hidden layers to effect each others' dynamics. Longer range dependencies are possible in both models but are omitted for simplicity.

6.2.3 CNN-FLDCRF

Convolutional Neural Networks (CNN) have been influential in several computer vision (CV) applications, viz., object recognition [34, 97] on the ImageNet data [98], semantic segmentation [35], action recognition [29, 99] on UCF-101 [94] and HMDB-51 [95] datasets etc. CNNs are often combined with RNNs and LSTMs for time-series modeling in CV tasks like action recognition [29, 100]. The convolutional layers learn relevant features x_t from the image at slice t (or slices $t - n$ to t for 3D CNNs) and the time-series model e.g., LSTM models the sequential relationship among the states/labels $\{y_t\}_{t=1:T}$ (or $\{h_t\}_{t=1:T}$), which are modeled together with the convolutional features.

We observe FLDCRF to outperform LSTM on several datasets and experiments. We are inclined to replace LSTM in end-to-end models by FLDCRF as the sequence

model (see Fig. 6.2), and intend to apply the resulting CNN-FLDCRF to standard action recognition tasks on the UCF-101 [94] and HMDB-51 [95] datasets. Due to several attractive modeling features of FLDCRF (see Chapter 3), we are hopeful that such CNN-FLDCRF models can be applied to real-life action recognition with stable, better and consistent performance.

Figure 6.2: A CNN-FLDCRF model.

6.2.4 Semi-supervised/Unsupervised Interaction Models

The FLDCRF interaction model can be tweaked to accommodate semi-labeled and unlabeled structures, as illustrated in Fig. 6.3.

In a social environment, labeling for all objects may not be necessary, or labels may not be available during training for some objects. For instance, in the pedestrian intention prediction problem, it is necessary to model the interaction among pedestrians and vehicles in the scene for early prediction, as we observed from our models in Chapters 4 and 5. However, we may not need to unnecessarily label the vehicles if we do not want such label outcomes during testing. Rather, we may associate hidden variable to capture vehicle latent-dynamics, with hidden variables assuming predefined sets of states. The transition among states capture different dynamic patterns of the vehicle. The FLDCRF model in Chapter 3 can be easily extended to accommodate such semi-supervised structures (see Fig. 6.3a) as follows:

$$P(\{\mathbf{y}_p\}_{1:N_p} \mid \mathbf{x}, \theta) = \sum_{\{\mathbf{h}_p\}_{1:N_p} : \forall h_{p,t} \in \mathcal{H}_{p,y_{p,t}}} \sum_{\{\mathbf{h}_v\}_{1:N_v}} P(\{\mathbf{h}_p\}_{1:N_p}, \{\mathbf{h}_v\}_{1:N_v} \mid \mathbf{x}, \theta). \quad (6.1)$$

Equation (6.1) assumes N_p pedestrians and N_v vehicles interacting with each other. The derivation is straightforward from the discussion in Chapter 3.4, and we leave it as a task for enthusiastic readers. Such a joint semi-supervised structure can model the temporal scene dynamics in general, and we may not need to add vehicle context information explicitly to the input feature x_t as in Chapters 4 and 5. Rather, the vehicle dynamics can be modeled along different layers while influencing pedestrians on-road in general.

Figure 6.3: Semi-labeled and Unlabeled FLDCRF interaction variants.

The idea of interacting hidden layers can be extended to unlabeled generic scene models. A simple version is illustrated in Fig. 6.3b. The model can be written as:

$$P(\{\mathbf{x}_p\}_{1:N_p}, \{\mathbf{x}_v\}_{1:N_v} \mid \theta) = \sum_{\{\mathbf{h}_p\}_{1:N_p}} \sum_{\{\mathbf{h}_v\}_{1:N_v}} P(\{\mathbf{x}_p\}_{1:N_p}, \{\mathbf{x}_v\}_{1:N_v}, \{\mathbf{h}_p\}_{1:N_p}, \{\mathbf{h}_v\}_{1:N_v} \mid \theta), \quad (6.2)$$

assuming N_p pedestrians and N_v vehicles interacting with each other. The model can be factorized for simpler computations according to the structure.

Such general unlabeled scene models $\hat{\theta}$ in Fig. 6.3b can be learned to solve several scene understanding problems, e.g., path prediction [7], anomaly detection [104, 105] etc. We address the inference tasks for such problems below.

Path Prediction

Trajectory estimation or path prediction is a very popular and important problem in Autonomous vehicle research.

As briefed in Chapter 4.3, [7], [18], [19], [22], [20], [24], [25] all consider the problem of path prediction, while [18], [22], [20], [24] and [25] take into account neighbouring variables in trajectory estimation task.

To extend our approach to pedestrian path prediction, we wish to include vehicle interaction context by the unsupervised general model (see Fig. 6.3b) to predict future position. In general, our task would be to frame a probabilistic distribution over possible future positions of a pedestrian given past position and motion features of him/her and other elements (static and dynamic contexts) in the scene. Then, we would need to find the most probable future position.

The possible structure of the model is depicted in Fig. 6.3b. A possible inference mechanism from the learned model $\hat{\theta}$ is described below. The inference method shown

here considers one pedestrian and one vehicle, i.e., replace $x_{p,2}$, $h_{p,2}$ by x_v , h_v in Fig. 6.3b.

- Inference Task:

Given a learned model $\hat{\theta}$, the goal would be to estimate future pedestrian positions $\hat{x}_{p,t+k}$ at a given k frame ahead given all (pedestrian/vehicle)s' current and past positions, i.e.

$$\hat{x}_{p,t+k} = \operatorname{argmax}_{x_{p,t+k}} P(x_{p,t+k} \mid \{x_{p,1:t}\}_{1:n_p}, \{x_{v,1:t}\}_{1:n_v}, \hat{\theta}) \quad (6.3)$$

The inference task can be carried out by using the following method:

$$\begin{aligned} P(x_{p,t+k} \mid x_{p,1:t}, x_{v,1:t}) &= \sum_{h_{p,t+k}} \sum_{h_{v,t+k}} P(x_{p,t+k}, h_{p,t+k}, h_{v,t+k} \mid x_{p,1:t}, x_{v,1:t}) \\ &= \sum_{h_{p,t+k}} \sum_{h_{v,t+k}} P(x_{p,t+k} \mid h_{p,t+k}) \cdot P(h_{p,t+k}, h_{v,t+k} \mid x_{p,1:t}, x_{v,1:t}) \end{aligned} \quad (6.4)$$

Saying $P(h_{p,t+k}, h_{v,t+k} \mid x_{p,1:t}, x_{v,1:t}) = \alpha_t(h_{p,t+k}, h_{v,t+k})$, we compute the following recursions:

$$\alpha_t(h_{p,t+k}, h_{v,t+k}) = \sum_{h_{p,t+k-1}} \sum_{h_{v,t+k-1}} P(h_{p,t+k}, h_{v,t+k}, h_{p,t+k-1}, h_{v,t+k-1} \mid x_{p,1:t}, x_{v,1:t}) \quad (6.5)$$

$$= \frac{1}{Z} \sum_{h_{p,t+k-1}} \sum_{h_{v,t+k-1}} \phi(h_{p,t+k}, h_{v,t+k}, h_{p,t+k-1}, h_{v,t+k-1}) \cdot P(h_{p,t+k-1}, h_{v,t+k-1} \mid x_{p,1:t}, x_{v,1:t}) \quad (6.6)$$

$$= \frac{1}{Z} \sum_{h_{p,t+k-1}} \sum_{h_{v,t+k-1}} \phi(h_{p,t+k}, h_{v,t+k}, h_{p,t+k-1}, h_{v,t+k-1}) \cdot \alpha_t(h_{p,t+k-1}, h_{v,t+k-1}), \quad (6.7)$$

Until we obtain $\alpha_t(h_{p,t}, h_{v,t})$ on the right hand side. $\alpha_t(h_{p,t}, h_{v,t}) = P(h_{p,t}, h_{v,t} \mid x_{p,1:t}, x_{v,1:t})$ can be obtained by the forward recursions (see Chapter 2.6) using model parameters.

We plan to test this model on standard Stanford [101] and ETH datasets [102].

Anomaly Detection

An abnormal event in a scene would be a deviation from natural dynamic patterns of different objects. Recent papers [104] and [105] deal with anomaly detection in video events by Neural Networks.

We describe a possible inference mechanism from the learned unsupervised model $\hat{\theta}$ (see Fig. 6.3b) for abnormal event prediction.

- Inference Task:

Any event in time $\mathbf{x}_{t_1:t_2} = \{\{x_p\}_{1:n_p}, \{x_v\}_{1:n_v}\}_{t_1:t_2}$, will be termed abnormal if the probability $P(\mathbf{x}_{t_1:t_2} | \theta)$ is below a given threshold. Computing $P(\mathbf{x}_{t_1:t_2} | \hat{\theta})$ is pretty straight forward as it requires us to estimate the joint probabilities:

$$\begin{aligned} P(\mathbf{x}_{t_1:t_2} | \hat{\theta}) &= \sum_{\mathbf{h}_{t_1:t_2}} P(\mathbf{x}_{t_1:t_2}, \mathbf{h}_{t_1:t_2} | \hat{\theta}) \\ &= \sum_{\mathbf{h}_{t_1:t_2}} \prod_{t=t_1}^{t_2} \phi_t(x_t, h_t, h_{t-1}), \end{aligned} \quad (6.8)$$

Where $\mathbf{h}_{t_1:t_2} = \{\{h_p\}_{1:n_p}, \{h_v\}_{1:n_v}\}_{t_1:t_2}$, $x_t = \{\{x_{p,t}\}_{1:n_{p_t}}, \{x_{v,t}\}_{1:n_{v_t}}\}$ and $h_t = \{\{h_{p,t}\}_{1:n_{p_t}}, \{h_{v,t}\}_{1:n_{v_t}}\}$. Equation (6.8) is a well known sum-product form of dynamic programming and can be computed according to model factorizations.

6.2.5 Heterogeneous state-space models

The factored state-space in FLDCRF can be utilized to capture interaction among heterogeneous state-spaces (see Fig. 6.4), i.e., interacting discrete and continuous state spaces. $h_{d,t}$ describes the discrete state space layer, while $h_{c,t}$ forms the continuous state-space layer. Such a model can be utilized for a joint intention and path prediction task for pedestrians, where the discrete layer models the latent-dynamic within the intention labels, and the continuous layer modeling the trajectory. The model is similar to a Switched Linear Dynamical Systems (SLDS [103]), only with the discrete state-space $h_{d,t}$ is derived from a supervised layer of class labels y_t .

Figure 6.4: A FLDCRF with heterogeneous interacting state space.

6.3 Future Research on the IP Model

Our aim is to build real-time working prototypes for our IP model. We will continue to train our model on bigger datasets and look forward to actual installations on AV and ADAS. For such installations, a head pose estimation [107] system also needs to be intergrated to the system in order to determine the context awareness of relevant pedestrians, in order to efficiently apply the context models.

We describe future research directions on the model below.

6.3.1 Pedestrian-pedestrian Interactions

With multiple pedestrians in front of the vehicle, pedestrian intentions often tend to vary in a group, i.e., the crossing/not-crossing intention of one pedestrian may depend on others. Therefore, we need to consider pedestrian-pedestrian interactions in our model for early and accurate prediction of the crossing behaviour. We have proposed FLDCRF-i in order to accommodate such interactions. Currently, we are applying FLDCRF-i (see Fig. 6.1) on the LI data [5] and hope to publish our results of such interaction models in the near future.

6.3.2 Augment by Static Context

As described in Chapter 1, static context variables, e.g., traffic lights, zebra crossings, bus stops etc. have been considered for pedestrian behaviour prediction in [8, 11, 12, 14]. Although these variables are not always present during navigation, it is important to consider such context when they are present for better prediction. We intend to augment our intention prediction system with such static context variables in the near future.

We depicted a complete context model for early pedestrian behaviour prediction in Fig. 1.5. We present our revised model for realizing this model in Fig. 6.5 as our future work. Other static context variables, e.g., number of lanes, area (parking lot, urban, rural etc.) can also be included provided sufficient training data.

6.3.3 End-to-end Context Models for IP

Convolutional Neural Networks (CNN) have been influential for automatically learning features to be fed to models. Usually, CNN layers are combined with LSTM layers (CNN-LSTM [29]) as end-to-end models that simultaneously learn the features and the temporal relationship among sequence values. We intend to replace our feature

Figure 6.5: Realizing the context model in Fig. 1.5 by revising our models. The revisions are shown in red. ‘TL’ stands for traffic lights, ‘ZC’ for zebra crossings, and ‘BS’ for bus stops.

extraction modules by such end-to-end models (e.g., CNN-LSTM, CNN-FLDCRF etc.) in the future (see Fig. 6.6) and compare performances.

Figure 6.6: An end-to-end context model for pedestrian intention prediction.

In addition, we would also like to extend our approach for intention prediction of other road users, such as cyclists and vehicles.

Our Publication(s)

1. S. Neogi, M. Hoy, C. Weng, J. Dauwels, “Context Based Pedestrian Intention Prediction Using Factored Latent Dynamic Conditional Random Fields”, In *Proc. of the IEEE Symposium Series on Computational Intelligence-2017*, Nov 27 - Dec 1, Hilton Hawaiian Village Wakiki Beach Resort, 2005 Kalia Rd, Honolulu, HI 96815, USA, 2017 [3].
2. S. Neogi, M. Hoy, K. Dang, H. Yu, J. Dauwels, “Context Model for Pedestrian Intention Prediction using Factored Latent-Dynamic Conditional Random Fields”. Accepted by *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*, available at: <https://arxiv.org/abs/1907.11881>, 2019.
3. S. Neogi, J. Dauwels, “Factored Latent-Dynamic Conditional Random Fields for Single and Multi-label Sequence Modeling”, Submitted to *Pattern Recognition*, available at: <https://arxiv.org/abs/1911.03667>, 2019.
4. S. Neogi, J. Dauwels, “Factored Latent-Dynamic Conditional Random Fields for Single and Multi-label Sequence Tagging”, Submitted to *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020.

Bibliography

- [1] S. Neogi, M. Hoy, K. Dang, H. Yu, J. Dauwels, “Context Model for Pedestrian Intention Prediction using Factored Latent-Dynamic Conditional Random Fields”. Accepted by *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*, available at: <https://arxiv.org/abs/1907.11881>, 2019.
- [2] S. Neogi, J. Dauwels, “Factored Latent-Dynamic Conditional Random Fields for Single and Multi-label Sequence Modeling”, Submitted to *Pattern Recognition*, available at: <https://arxiv.org/abs/1911.03667>, 2019.
- [3] S. Neogi, M. Hoy, C. Weng, J. Dauwels, “Context Based Pedestrian Intention Prediction Using Factored Latent Dynamic Conditional Random Fields”. In *Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, DOI: 10.1109/SSCI.2017.8280970, 2017.
- [4] S. Neogi, J. Dauwels, M. Hoy, “NTU Pedestrian Dataset”. Figshare. Dataset. <https://doi.org/10.6084/m9.figshare.11455545.v1>, 2019.
- [5] S. Neogi, J. Dauwels, P. Tan, “Little India Dataset”, *data processing in progress*, 2020.
- [6] D. Koller and N. Friedman (2009). “Probabilistic Graphical Models: Principles and Techniques”, *MIT Press*, 2009.
- [7] C. G. Keller, D.M. Gavrilu, “Will the Pedestrian Cross? A Study on Pedestrian Path Prediction”, *IEEE Transactions On Intelligent Transportation Systems*, VOL. 15, NO. 2, APRIL 2014.
- [8] F. Schneemann, P. Heinemann, “Context-based Detection of Pedestrian Crossing Intention for Autonomous Driving in Urban Environments”. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2016.
- [9] A. Schulz and R. Stiefelhagen, “Pedestrian Intention Recognition using Latent-dynamic Conditional Random Fields” *2015 IEEE Intelligent Vehicles Symposium (IV)*, 2015.

- [10] S. Köhler, M. Goldhammer, S. Bauer, S. Zecha, K. Doll, U. Brunsmann, K. Dietmayer, “Stationary Detection of the Pedestrian’s Intention at Intersections”, *IEEE Intelligent Transportation Systems Magazine*, Volume: 5, Issue: 4, winter 2013.
- [11] J. Hariyono and K. H. Jo, “Pedestrian Action Recognition using Motion Type Classification”, *2015 IEEE 2nd International Conference on Cybernetics (CYBCONF)*, 2015.
- [12] Y. Hashimoto, Y. Gu, L. Hsu, M. Iryo-Asano, S. Kamijo, “A Probabilistic Model of Pedestrian Crossing Behavior at Signalized Intersections for Connected Vehicles”, *Transportation Research Part C: Emerging Technologies*, Volume 71, October 2016, Pages 164-181.
- [13] R. Quintero, I. Parra, J. Lorenzo, D. Fernandez-Llorca and M. A. Sotelo, “Pedestrian Intention Recognition by Means of a Hidden Markov Model and Body Language”, *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017.
- [14] A. Rasouli, I. Kotseruba, J. K. Tsotsos. “Are They Going to Cross? A Benchmark Dataset and Baseline for Pedestrian Crosswalk Behavior”. In *Proc. of the International Conference on Computer Vision Workshop (ICCVW)*, 2017.
- [15] P. Gujjar, R. Vaughn, “Classifying Pedestrian Actions In Advance Using Predicted Video of Urban Driving Scenes”. In *Proc. of the International Conference on Robotics and Automation (ICRA)*, 2019.
- [16] Z. Fang, D. Vázquez, A. M. López, “On-Board Detection of Pedestrian Intentions”, *Sensors*, 2017, doi: 10.3390/s17102193.
- [17] M. Hoy, Z. Tu, D. Kang, and J. Dauwels (2018), “Learning to Predict Pedestrian Intention via Variational Tracking Networks”. In *Proceedings of the IEEE 21st International Conference on Intelligent Transportation Systems*, Maui, USA [accepted].
- [18] J. F. P. Kooij, N. Schneider, F. Flohr, D. M. Gavrilu, “Context-Based Pedestrian Path Prediction”. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2014.
- [19] A. T. Schulz, R. B. GmbH, “A Controlled Interactive Multiple Model Filter for combined Pedestrian Intention Recognition and Path Prediction”, *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 2015.

- [20] V. Karasev, A. Ayvaci, B. Heisele, S. Soatto, “Intent-Aware Long-Term Prediction of Pedestrian Motion”. In *Proc. of the International Conference on Robotics and Automation (ICRA)*, 2016.
- [21] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, S. Savarese, “Social LSTM: Human Trajectory Prediction in Crowded Spaces”. In *Proc. of the Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [22] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, M. Chandraker, “DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents”. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [23] I. Kotseruba, A. Rasouli, J. K. Tsotsos, “Joint Attention in Autonomous Driving (JAAD).” arXiv preprint arXiv:1609.04741 (2016).
- [24] D. Varshneya, G. Srinivasaraghavan, “Human Trajectory Prediction using Spatially aware Deep Attention Models”. In *Proc. of the Neural Information Processing Systems (NIPS)*, 2017.
- [25] W. C. Ma, D. Huang, N. Lee, K. M. Kitani, “Forecasting Interactive Dynamics of Pedestrians with Fictitious Play”. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [26] J.F.P. Kooij, F. Flohr, E.A.I. Pool, D. M. Gavrila. “Context-Based Path Prediction for Targets with Switching Dynamics”. *International Journal of Computer Vision*, vol. 127, nr. 3, pp. 239-262, 2019.
- [27] E.A.I. Pool, J.F.P. Kooij, D. M. Gavrila. “Context-based cyclist path prediction using Recurrent Neural Networks”. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, Paris (France), 2019.
- [28] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, K. O. Arras. “Human motion trajectory prediction: A survey”. Accepted for *International Journal of Robotics Research (IJRR)*, also arXiv preprint arXiv:1905.06113 (2019).
- [29] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, S. W. Baik, “Action Recognition in Video Sequences using Deep Bi-directional LSTM with CNN features”, *IEEE Access*, 2017.
- [30] G. D. Forney Jr. “The Viterbi algorithm”. In *Proceedings of the IEEE*. 61 (3): 268–278. doi:10.1109/PROC.1973.9030, (March 1973).

- [31] B. Carpenter, D. Lee, M. A. Brubaker, A. Riddell, A. Gelman, B. Goodrich, J. Guo, M. Hoffman, M. Betancourt, P. Li, “Stan: A Probabilistic Programming Language”, *Journal of Statistical Software*, Volume VV, Issue II, Nov. 2017.
- [32] C. Liu. “Beyond Pixels: Exploring New Representations and Applications for Motion Analysis”. *Doctoral Thesis. Massachusetts Institute of Technology*. May 2009.
- [33] J. Hariyono, L. Kurnianguro, Wahyono, D. C. Hernandez , KH. Jo, “Ego-Motion Compensated for Moving Object Detection in a Mobile Robot”, In: Ali M., Pan JS., Chen SM., Horng MF. (eds) *Modern Advances in Applied Intelligence. IEA/AIE 2014. Lecture Notes in Computer Science*, vol 8482. Springer, Cham, 2014.
- [34] S. Ren, K. He, R. Girshick, J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In *Proc. of the Neural Information Processing Systems (NIPS)*, 2015.
- [35] A. Kendall, V. Badrinarayanan, R. Cipolla, “Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding”. In *Proc. of the The British Machine Vision Conference (BMVC)*, 2017.
- [36] MATLAB 8.0 and Statistics Toolbox 8.1, The MathWorks, Inc., Natick, Massachusetts, United States.
- [37] N. Schneider and D. M. Gavrila, “Pedestrian Path Prediction with Recursive Bayesian Filters: A Comparative Study”, In *Lecture Notes in Computer Science: Proc. of the German Conference on Pattern Recognition (GCPR)*, vol. 8142, Springer, 2013.
- [38] CMU, “CMU graphics lab motion capture database”, <http://mocap.cs.cmu.edu/subjects.php>.
- [39] F. Sha and F. Pereira. “Shallow parsing with conditional random fields”. In *Conference on Human Language Technology and North American Association for Computational Linguistics (HLT-NAACL)*, pages 213-220, 2003.
- [40] L. R. Rabiner. “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”. In *Proceedings of the IEEE*, Volume: 77, Issue: 2, Feb 1989, DOI: 10.1109/5.18626.

- [41] C. Sutton and A. McCallum, “An Introduction to Conditional Random Fields”, URL:<https://homepages.inf.ed.ac.uk/csutton/publications/crftutv2.pdf>
- [42] J. Lafferty, A. McCallum, F. C. N. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data”. In *Proceedings of the Eighteenth International Conference on Machine Learning*, Pages 282-289, June 2001.
- [43] S. Sarawagi, W. W. Cohen, . “Semi-Markov Conditional Random Fields for Information Extraction”. In Lawrence K. Saul, Yair Weiss, Léon Bottou (eds.). (2005). *Advances in Neural Information Processing Systems 17*. Cambridge, MA: MIT Press. pp. 1185–1192.
- [44] C. Sutton, K. Rohanimanesh, A. McCallum, “Dynamic Conditional Random Fields: Factorized Probabilistic Models for Labeling and Segmenting Sequence Data”. *Journal of Machine Learning Research*, 8:693-723, 2007.
- [45] L. P. Morency, A. Quattoni, T. Darrell, “Latent-Dynamic Discriminative Models for Continuous Gesture Recognition”. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, 2007.
- [46] L. R. Rabiner. “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [47] Z. Ghahramani, M. I. Jordan, “Factorial Hidden Markov Models”, *Machine Learning (1997)* 29: 245. <https://doi.org/10.1023/A:1007425814087>.
- [48] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, “Neural architectures for named entity recognition”. *arXiv:1603.01360*. [Online]. Available: <https://arxiv.org/abs/1603.01360>, March 2016.
- [49] F. J. Ordóñez, D. Roggen, “Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition”. *Sensors* 16(1), 115; <https://doi.org/10.3390/s16010115>, 2016.
- [50] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. “Natural language processing (almost) from scratch”. *Journal of Machine Learning Research*, 12:2493–2537, 2011.
- [51] K. P. Murphy, “Dynamic bayesian networks: representation, inference and learning”. *Doctoral Dissertation*, University of California, Berkeley, 2002.

- [52] A. McCallum, D. Freitag, F. C. N. Pereira, “Maximum Entropy Markov Models for Information Extraction and Segmentation”. In *Proceedings of the Seventeenth International Conference on Machine Learning*, Pages 591-598, 2000.
- [53] Z. Huang, W. Xu, and K. Yu, “Bidirectional LSTM-CRF models for sequence tagging”. CoRR, abs/1508.01991, 2015.
- [54] A. Quattoni, S. B. Wang, L. P. Morency, D. Demirdjian, T. Darrell, “Hidden Conditional Random Fields for Gesture Recognition”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, Volume: 29, Issue: 10, Oct. 2007.
- [55] X. Sun, L. P. Morency, D. Okanohara, J. Tsujii, “Modeling latent-dynamic in shallow parsing: a latent conditional model with improved inference”. In *Proceedings of the 22nd International Conference on Computational Linguistics*, Volume 1, Pages 841-848, 2008.
- [56] Z. Ghahramani, M. I. Jordan, “Factorial Hidden Markov Models”. *Machine Learning*, 29: 245. <https://doi.org/10.1023/A:1007425814087>, 1997.
- [57] D. Thai, S. H. Ramesh, S. Murty, L. Vilnis, A. McCallum, “Embedded-State Latent Conditional Random Fields for Sequence Labeling”, In *Proceedings of CoNLL 2018*, 2018.
- [58] S. Hochreiter and J. Schmidhuber. “Long short-term memory”. *Neural computation*, 9(8):1735–1780, 1997.
- [59] H. Zhu, I. Ch. Paschalidis, A. M. Tahmasebi, ”Context-based bidirectional-LSTM model for sequence labeling in clinical reports”. In *Proceeding of SPIE 10954, Medical Imaging 2019: Imaging Informatics for Healthcare, Research, and Applications*, DOI: 10.1117/12.2512103, March 2019.
- [60] R. Caruana. “Multitask Learning”. *Machine Learning*, 28(1):41–75, 1997.
- [61] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, W. Xu, “CNN-RNN: A Unified Framework for Multi-label Image Classification”. *arXiv:1604.04573*, 2016.
- [62] Y. Wei, W. Xia, M. Lin, J. Huang, B. Ni ; J. Dong, Y. Zhao, S. Yan, “HCP: A Flexible CNN Framework for Multi-Label Image Classification”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume: 38 , Issue: 9, Pages 1901 - 1907, 2015.

- [63] M. J. Choi, A. Torralba, A. S. Willsky, “A Tree-Based Context Model for Object Recognition”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, Volume: 34 , Issue: 2, Pages 240-252, 2012.
- [64] B. X. Nie, C. Xiong and S. C. Zhu, “Joint Action Recognition and Pose Estimation From Video”. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, DOI: 10.1109/CVPR.2015.7298734, 2015.
- [65] S. Changpinyo, H. Hu, F. Sha, “Multi-Task Learning for Sequence Tagging: An Empirical Study”. *arXiv:1808.04151*, 2018.
- [66] V. Pahuja, , A. Laha, , S Mirkin, , V. Raykar, L. Kotlerman, G. Lev, “Joint Learning of Correlated Sequence Labeling Tasks Using Bidirectional Recurrent Neural Networks”. In *Proceedings of the INTERSPEECH 2017*, 2017.
- [67] Z. Li, J. Chao, M. Zhang, W. Chen, “Coupled Sequence Labeling on Heterogeneous Annotations: POS Tagging as a Case Study”. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, DOI: 10.3115/v1/P15-1172, 2015.
- [68] G. Luo, X. Huang, C. Y. Lin, Z Nie, “Joint Named Entity Recognition and Disambiguation”. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888, 2015.
- [69] M. Dredze, P. P. Talukdar, K. Crammer, “Sequence Learning from Data with Multiple Labels”. In *Proceedings of the ECML-PKDD 2009 workshop on learning from multi-label data*, MLD, 2009.
- [70] Y. Shi, M. Wang, “A Dual-layer CRFs Based Joint Decoding Method for Cascaded Segmentation and Labeling Tasks”. In *Proceedings of the International Joint Conference on Artificial Intelligence 2007*, 2007.
- [71] E. F. T. K. Sang, S. Buchholz, “Introduction to the CoNLL-2000 Shared Task: Chunking”. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 127-132, Lisbon, Portugal, 2000.
- [72] E. F. T. K. Sang, F. D. Meulder, “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition”. In *CONLL '03 Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, Volume 4, Pages 142-147, Edmonton, Canada, 2003.

- [73] R. Chavarriaga, H. Sagha, A. Calatroni, S. Digumarti, G. Tröster, J. D. R. Millán, D Roggen. "The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition". *Pattern Recognition Letters*, 2013.
- [74] Y. Song, L. P. Morency, R. Davis, "Multi-View Latent Variable Discriminative Models For Action Recognition". In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2012*, 2012.
- [75] W. Lu and H. T. Ng, "Better Punctuation Prediction with Dynamic Conditional Random Fields". In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 177–186, 2010.
- [76] R. C. B. Madeo, C. A. M. Lima, S. M. PERES, "Gesture Unit Segmentation using Support Vector Machines: Segmenting Gestures from Rest Positions". In *Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC)*, p. 46-52, 2013.
- [77] G. D. Forney Jr. "The Viterbi algorithm". In *Proceedings of the IEEE*. 61 (3): 268–278. doi:10.1109/PROC.1973.9030, March, 1973.
- [78] P. K. Wagner, S. M. Peres, C. A. M. Lima, F. A. Freitas, R. C. B. Madeo, "Gesture Unit Segmentation Using Spatial-Temporal Information and Machine Learning". In *Proceedings of the Twenty-Seventh International Florida Artificial Intelligence Research Society Conference*, 2014.
- [79] Thomas W, "Thomas' answer to the question 'How to select number of hidden layers and number of memory cells in an LSTM?' posted at Artificial Intelligence Stack Exchange website". URL: <https://ai.stackexchange.com/questions/3156/how-to-select-number-of-hidden-layers-and-number-of-memory-cells-in-an-lstm>, 2017.
- [80] Multiple users, "Answers to the question 'How to choose the number of hidden layers and nodes in a feedforward neural network?' posted at Stack Exchange website". URL: <https://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw/1097#1097>, 2018-2019.
- [81] K. Eckhardt, "Choosing the right Hyperparameters for a simple LSTM using Keras", URL: <https://towardsdatascience.com/choosing-the-right-hyperparameters-for-a-simple-lstm-using-keras-f8e9ed76f046>, 2018.

- [82] M. Wollmer, M. Kaiser, F. Eyben, B. Schuller, G. Rigoll, “LSTM-Modeling of continuous emotions in an audiovisual affect recognition framework”. *Image and Vision Computing*, 31 (2013) 153–163, 2013.
- [83] J. Huang, W. Zhou, Q. Zhang, H. Li, W. Li, “Video-Based Sign Language Recognition without Temporal Segmentation”. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018.
- [84] A. Zadeh, P. P. Liang, S. Poria, Prateek Vij, E. Cambria, L. P. Morency, “Multi-Attention Recurrent Network for Human Communication Comprehension”. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018.
- [85] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, J. Schmidhuber, “LSTM: A Search Space Odyssey”. In *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS*, Volume: 28, Issue: 10, Pages: 2222 - 2232, DOI: 10.1109/TNNLS.2016.2582924, Oct. 2017.
- [86] Chuanhai Dong, Jiajun Zhang, Chengqing Zong, Masanori Hattori, Hui Di less, “Character-Based LSTM-CRF with Radical-Level Features for Chinese Named Entity Recognition”. In *Proc. of the ICCPOL 2016, NLPCC 2016*. Lecture Notes in Computer Science, vol 10102. Springer, 2016
- [87] LSTM, Wikipedia, *URL* : [https : //en.wikipedia.org/wiki/Long_short – term_memory](https://en.wikipedia.org/wiki/Long_short-term_memory).
- [88] Stack Overflow, LSTM network, *URL* : [https : //stackoverflow.com/questions/52647115/python – implementing – an – lstm – network – with – keras – and – tensorflow](https://stackoverflow.com/questions/52647115/python-implementing-an-lstm-network-with-keras-and-tensorflow).
- [89] Accident study, Absent-minded drivers, *URL* : [https : //www.smh.com.au/national/absentminded – drivers – to – blame – for – crashes – 20080820 – 3ylr.html](https://www.smh.com.au/national/absentminded-drivers-to-blame-for-crashes-20080820-3ylr.html), 2008.
- [90] Accident study 2, Absent-minded drivers, *URL* : amta.org.au, 2018.
- [91] Waymo Pedestrian Model, *URL* : [https : //www.youtube.com/watch?v = 4V – rxFxz – rs](https://www.youtube.com/watch?v=4V-rxFxz-rs), 2018.
- [92] Uber Pedestrian Accident, *URL* : [https : //www.theguardian.com/technology/video/2018/mar/22/in – car – footage – shows – fatal – self – driving – crash – video](https://www.theguardian.com/technology/video/2018/mar/22/in-car-footage-shows-fatal-self-driving-crash-video), 2019.

- [93] Perceptive Automata, <https://www.perceptiveautomata.com/>, 2018.
- [94] K. Soomro, A. R. Zamir and M. Shah, “UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild”, *CRCV-TR-12-01*, November, 2012.
- [95] H. Jhuang, J. Gall, S. Zuffi, C. Schmid and M. J. Black, “Towards understanding action recognition”, in *proceedings of International Conf. on Computer Vision (ICCV)*, pages 3192-3199, 2013.
- [96] A. Efentakis, S. Brakatsoulas, N. Grivas, G. Lamprianidis, K. Patroumpas, D. Pfoser, “Towards a Flexible and Scalable Fleet Management Service”, *IWCTS '13 Proceedings of the Sixth ACM SIGSPATIAL International Workshop on Computational Transportation Science*, 2013.
- [97] A. Krizhevsky, I. Sutskever and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, in *proceedings of NIPS*, 2012.
- [98] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. “ImageNet: A Large-Scale Hierarchical Image Database”, in *proceedings of CVPR*, 2009.
- [99] J. Carreira, A. Zisserman, “Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset”, *URL: <https://arxiv.org/abs/1705.07750>*, 2017.
- [100] F. J. Ordonez and D. Roggen, “Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition”, *Sensors*, 2016.
- [101] A. Robicquet, A. Sadeghian, A. Alahi, S. Savarese, “Learning Social Etiquette: Human Trajectory Prediction In Crowded Scenes”. In *European Conference on Computer Vision (ECCV)*, 2016.
- [102] B. Leibe, N. Cornelis, K. Cornelis, L. V. Gool, “Dynamic 3D Scene Analysis from a Moving Vehicle”. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, 2007.
- [103] S. W. Linderman, A. C. Miller, R. P. Adams, D. M. Blei, L. Paninski, M. J. Johnson, “Recurrent switching linear dynamical systems”, <https://arxiv.org/abs/1610.08466>, 2016.
- [104] Y. S. Chong, Y. H. Tay, “Abnormal Event Detection in Videos using Spatiotemporal Autoencoder”. In *International Symposium on Neural Networks*, 2017.
- [105] M. Ravanbakhsh, M. Nabi, E. Sangineto, L. Marcenaro, C. Regazzoni, N. Sebe, “Abnormal Event Detection in Videos using Generative Adversarial Nets”. In

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition ICIP 2017.

- [106] M. J. Choi, “Trees and Beyond: Exploiting and Improving Tree-structured Graphical Models”, *PhD Thesis*, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 2011.
- [107] A. Schulz, N. Damer, M. Fischer, and R. Stiefelhagen, “Combined head localization and head pose estimation for video based advanced driver assistance systems.” In *Proceedings of Pattern Recognition (DAGM)*, 2011.