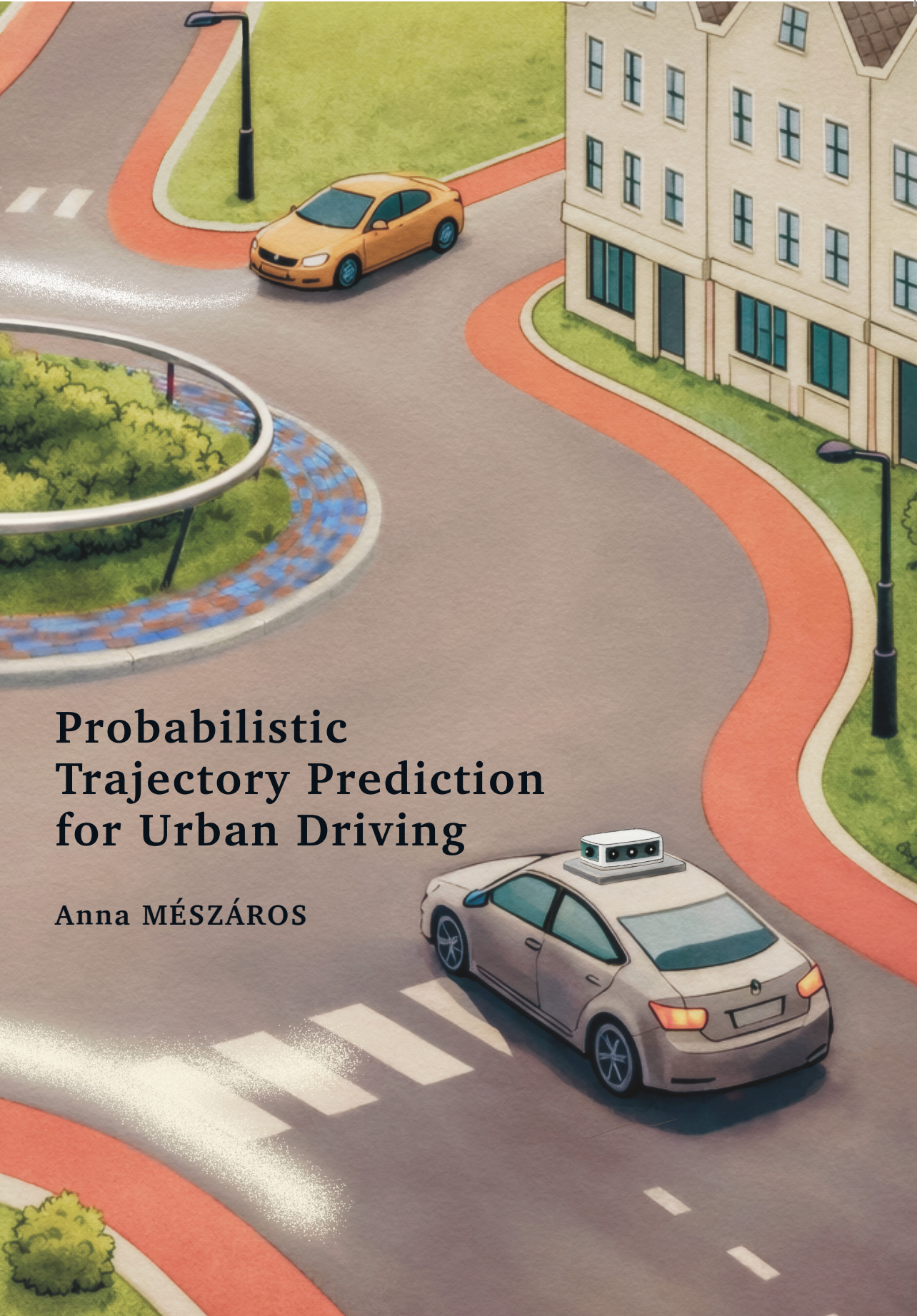


Probabilistic Trajectory Prediction for Urban Driving

Anna MÉSZÁROS

Probabilistic Trajectory Prediction for Urban Driving

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus Prof. dr. ir. H. Bijl,
chair of the Board of Doctorates,
to be defended publicly on
Thursday, 9 April 2026 at 10:00

by

Anna MÉSZÁROS

This dissertation has been approved by the promoters.

Composition of the doctoral committee:

Rector Magnificus,	*Chairperson*
Prof. Dr.-Ing. J. Kober,	Delft University of Technology & University of Stuttgart, *promotor*
Prof. Dr. J. Alonso-Mora,	Delft University of Technology, *promotor*
Independent Members:	
Prof.dr.ir. J.C.F. de Winter,	Delft University of Technology
Assoc. prof. A. Alahi,	Swiss Federal Institute of Technology in Lausanne
Assoc. prof. F. García Fernandez,	Charles III University of Madrid
Dr. J.F.P. Kooij,	Delft University of Technology
Prof.dr.ir. R. Happee,	Delft University of Technology, *reserve member*

The work in the thesis was funded by the project "Acting Under Uncertainty" of the Netherlands Organization for Scientific Research (NWO), Dutch Research Agenda (NWA), NWA.1292.19.298.

Keywords: Human Trajectory Prediction, Probabilistic Trajectory Prediction, Density Estimation, Autonomous Vehicles, Urban Driving

Printed by: Ridderprint | www.ridderprint.nl

Style: TU Delft House Style, with modifications by Moritz Beller
`https://github.com/Inventitech/phd-thesis-template`

The author set this thesis in LaTeX using the Libertinus and Inconsolata fonts.

ISBN/EAN: 978-94-6384-930-2

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

We are stuck with technology when what we really want is just stuff that works.

-Douglas Adams

Contents

Summary

Driving in urban environments is a challenging task, even for us humans. There is a variety of different traffic participants including other drivers, cyclists and pedestrians who each have their own unique behaviors. In order to navigate around them safely it is not enough to have detected their presence. We also need to reason about what they might do. However, we cannot always be certain of others' course of action as we do not know which route they are taking, how risk-averse they are or how aware they are of their surroundings, along with a number of other psychological factors. Hence, we also need to reason how likely it is that they will take a particular course of action. For autonomous vehicles to navigate in urban environments with other traffic participants, we need to imbue them with the capacity to reason about what other participants may do. The main goal of this thesis is to provide contributions in the development of probabilistic trajectory prediction models which accurately capture the distribution over possible future trajectories of other traffic participants.

The trajectories humans may take exhibit uncertainty on the high-level route intentions of a person (i.e. the modes) as well as low-level variability in how they execute maneuvers. To capture all these possibilities and attribute a likelihood, we need sophisticated and reliable density estimators. These density estimators can then enable autonomous vehicles (AVs) to reason about the intentions of other people in traffic and plan around them. However, density estimators are not only at the core of probabilistic prediction models, but they are also relevant for developing more reliable evaluation frameworks of such probabilistic models. After a brief introduction of the problematic at hand, this thesis begins by investigating density estimation over complex, high-dimensional, multi-modal distributions. Kernel Density Estimation (KDE), is perhaps one of the most popular density estimators with it being used in a number of applications, including the evaluation of probabilistic prediction models. Despite its popularity, KDE has a strong tendency to over-smooth multi-modal distributions. Even more recent state-of-the-art density estimators suffer from over-fitting or over-smoothing on non-Gaussian multi-modal distributions, and in some cases a complete inability to estimate the underlying distribution when the dimensionality of the distribution is increased. We propose a non-parametric density estimator which addresses these shortcomings. Our RObust Multi-modal Estimator (ROME) utilizes clustering to segment multi-modal samples into multiple uni-modal ones. The data in each cluster is then decorrelated and standardized before calculating the KDE for these individual clusters. The density estimates of the individual clusters are then combined into a single multi-modal estimate, which better captures non-Gaussian, high-dimensional, multi-modal distributions compared to the state-of-the-art. This improved density estimation plays a key role for a more reliable evaluation of probabilistic prediction models.

Now equipped with a better means to evaluate probabilistic prediction models, we next investigate marginal trajectory prediction for urban driving participants, with a focus

on improving the fit of the predicted distribution. We focus specifically on predicting distributions over trajectories, as the commonly used approach of predicting distributions per timestep leads to more conservative motion planning. However, effectively learning a multi-modal distribution is challenging, and a number of models continue to suffer from mode collapse or predicting highly uncertain distributions. To this end we leverage Normalizing Flows (NFs), a class of generative methods geared towards distribution fitting with exact likelihood computation. We further establish that by learning a distribution over an abstraction of the trajectories – obtained with an autoencoder – we are able to achieve a better distribution fit.

Given the interactive nature of human navigation in urban environments, marginal distributions – i.e. distributions over the actions of individual traffic participants instead of all traffic participants simultaneously – are generally not enough to capture the true evolution of a traffic scene. People are constantly negotiating and adjusting their own actions in interaction with other traffic participants to ensure the safety of everyone involved, which impacts everyone's actions in the future. With marginal distributions, certain futures – such as a collision between two vehicles – can thus appear far more likely than they are in reality. For this reason, it is relevant to predict the joint distribution over the trajectories of all traffic participants in a scene. An important aspect of making such predictions is the representation of the interactions themselves. A number of state-of-the-art methods rely on Graph Neural Networks (GNNs) to establish the interactive links between traffic participants in a scene. Our investigation, however, indicates that relying on neural networks to extract this information from data can have a detrimental effect on the predicted distribution. Instead, we show that using representations based on human reasoning in order to guide the learning of these interactions has the potential to improve performance. While these findings highlight the importance of modeling interactions effectively, they also raise questions about how well current evaluation metrics reflect a model's ability to capture such complex, multi-modal behaviors.

To conclude, we look into the evaluation of joint trajectory prediction models from the perspective of mode collapse. To this end we propose an evaluation framework on the basis of homotopy classes, with focus on safety-critical settings where traffic participants' paths may cross or influence each other. Within this framework we introduce metrics for quantifying mode collapse, mode correctness, and mode coverage. We additionally consider how the predictions change over time so as to detect behaviors like switching the most likely mode from one prediction to the next. We observe that existing evaluation metrics for joint trajectory prediction —such as minADE/FDE, or most likely ADE/FDE — fail to meaningfully assess whether a model actually captures all relevant modes of a joint distribution. In experiments on several joint trajectory prediction models, we show that mode collapse does occur and that in some cases models are not able to predict the correct mode even when the interaction becomes nearly inevitable.

In conclusion, this thesis addresses the problem of probabilistic trajectory prediction on different levels of the problem. It advances density estimation of complex, multi-modal distributions, which is relevant not only for better evaluation of probabilistic trajectory prediction models but is also applicable to a wide range of problems across different domains. It further investigates how to improve the structure of probabilistic prediction models for both marginal and joint distributions. Lastly, it delves into the problem of evaluating

probabilistic prediction models and proposes an evaluation framework for evaluating the predicted modes of joint distributions. These contributions enhance AVs' ability to reason about human intent and uncertainty, enabling safer and more informed decision making.

Beyond improving safety and reliability, the proposed approaches also offer broader societal benefits. More accurate trajectory prediction can lead to smoother traffic flow, reducing congestion and associated carbon emissions. The developed methodologies further align with smart city initiatives by enabling predictive coordination of AVs, facilitating data-driven mobility and more efficient use of urban transport networks. Additionally, by supporting safer and more predictable navigation, these advancements can improve accessibility for elderly and disabled individuals. Together, these contributions not only advance the field of probabilistic trajectory prediction but also promote more sustainable, inclusive and intelligent transportation systems.

Samenvatting

Autorijden in stedelijke omgevingen is een uitdagende taak. Er is een grote verscheidenheid aan verkeersdeelnemers, waaronder automobilisten, fietsers en voetgangers, die elk hun eigen unieke gedrag vertonen. Om veilig tussen deze verkeersdelnemers door te navigeren, is het niet voldoende om enkel hun aanwezigheid te detecteren; we moeten ook kunnen redeneren over wat zij mogelijk zullen doen. Echter, we kunnen niet altijd zeker zijn van andermans handelwijze, omdat we niet weten welke route zij nemen, hoe risicomijdend zij zijn of hoe bewust zij zich zijn van hun omgeving, naast een reeks andere psychologische factoren. Daarom moeten we ook kunnen inschatten hoe waarschijnlijk het is dat iemand een bepaalde handeling zal uitvoeren. Voor autonome voertuigen (AV's) die zich in stedelijke omgevingen tussen andere verkeersdeelnemers bewegen is het dus noodzakelijk dat zij het vermogen hebben om te redeneren over wat anderen zouden kunnen doen. Het hoofddoel van deze thesis is dan ook om bij te dragen aan de ontwikkeling van probabilistische trajectvoorspellingsmodellen die nauwkeurig de verdeling over mogelijke toekomstige trajecten van andere verkeersdeelnemers kunnen vastleggen.

De trajecten die mensen kunnen nemen vertonen zowel onzekerheid over de globale route-intenties van een persoon (de modi) als de variabiliteit op laag niveau in de manier waarop zij maneouvres uitvoeren. Om al deze mogelijkheden vast te leggen en er kanswaarden aan toe te kennen, hebben we geavanceerde en betrouwbare dichtheidsschatters nodig. Deze dichtheidsschatters stellen autonome voertuigen (AV's) vervolgens in staat om te redeneren over de intentiesvan anderen in het verkeer en hun eigen bewegingen hieromheen te plannen. Dichtheidsschatters vormen echter niet alleen de kern van probabilistische voorspellingsmodellen, maar zijn ook relevant voor de ontwikkeling van betrouwbaardere evaluatiekaders voor dergelijke probabilistische modellen. Na een korte inleiding tot het probleem onderzoekt deze thesis dichtheidsschatting over complexe, hoog-dimensionale, multimodale verdelingen. Kernel Density Estimation (KDE) is een van de populairste dichtheidsschatters en wordt in verschillende toepassingen gebruikt, waaronder de evaluatie van probabilistische voorspellingsmodellen. Ondanks zijn populariteit heeft KDE echter de neiging om multimodale verdelingen te sterk te over-smoothing. Zelfs recente, geavanceerde dichtheidsschatters lijden onder over-fitting of over-smoothing bij niet-Gaussiaanse, multimodale verdelingen, en falen soms volledig wanneer de dimensionaliteit van de verdeling toeneemt. Wij stellen een niet-parametrische dichtheidsschatting voor die deze tekortkomingen aanpakt. Onze RObust Multi-modal Estimator (ROME) maakt gebruik van clustering om multimodale gegevens op te splitsen in meerdere unimodale clusters. De data in elk cluster wordt vervolgens gedecorreleerd en gestandaardiseerd voordat de KDE per cluster wordt berekend. De dichtheidsschattingen van de afzonderlijke clusters worden daarna gecombineerd tot één multimodale schatting, die niet-Gaussiaanse, hoog-dimensionale, multimodale verdelingen beter vastlegt dan de huidige state-of-the-art. Deze verbeterde dichtheidsschatting speelt een sleutelrol bij een betrouwbaardere evaluatie van probabilistische voorspellingsmodellen.

Met een verbeterde methode om probabilistische voorspellingsmodellen te evalueren, richten we ons vervolgens op marginale trajectvoorspelling voor verkeersdeelnemers in stedelijke omgevingen, met nadruk op het verbeteren van de nauwkeurigheid van de voorspelde verdeling. We concentreren ons specifiek op het voorspellen van verdelingen over volledige trajecten, aangezien de gangbare aanpak, het voorspellen per tijdstap, vaak leidt tot te conservatieve bewegingsplanning. Het effectief leren van een multimodale verdeling is echter uitdagend, en veel modellen lijden nog steeds aan mode collapse of voorspellen zeer onzekere verdelingen. Hiervoor maken we gebruik van Normalizing Flows (NF's), een klasse van generatieve methoden die zijn ontworpen voor nauwkeurige "distribution fitting" met exacte waarschijnlijkheidsberekening. We tonen bovendien aan dat het leren van een verdeling over een abstractie van de trajecten – verkregen met een autoencoder – leidt tot een betere distribution fitting.

Gezien de interactieve aard van menselijk rijgedrag in stedelijke omgevingen, zijn marginale verdelingen – d.w.z. verdelingen over de acties van individuele verkeersdeelnemers in plaats van alle verkeersdeelnemers gezamenlijk – doorgaans niet voldoende om de werkelijke ontwikkeling van een verkeerssituatie vast te leggen. Mensen onderhandelen voortdurend en passen hun gedrag aan in interactie met andere verkeersdeelnemers om de veiligheid van iedereen te waarborgen, wat ieders toekomstige acties beïnvloedt. Met marginale verdelingen kunnen daardoor bepaalde toekomstscenario's – zoals een botsing tussen twee voertuigen – veel waarschijnlijker lijken dan ze in werkelijkheid zijn. Daarom is het relevant om de gezamenlijke verdeling over de trajecten van alle verkeersdeelnemers in een scenario te voorspellen. Een belangrijk aspect van dergelijke voorspellingen is de representatie van de interacties zelf. Veel toonaangevende methoden maken gebruik van Graph Neural Networks (GNN's) om de interacties tussen verkeersdeelnemers in een scène te modelleren. Ons onderzoek toont echter aan dat het vertrouwen op neurale netwerken om deze informatie rechtstreeks uit data te leren een negatief effect kan hebben op de kwaliteit van de voorspelde verdeling. In plaats daarvan laten wij zien dat representaties gebaseerd op menselijke redenering, die het leerproces van deze interacties begeleiden, de prestaties kunnen verbeteren. De resultaten benadrukken niet alleen de noodzaak van een nauwkeurige modellering van interacties, maar brengen ook twijfel naar voren over de mate waarin bestaande evaluatiemaatstaven het vermogen van modellen adequaat beoordelen om complexe, multimodale gedragingen te representeren.

Tot slot richten we ons op de evaluatie van gezamenlijke trajectvoorspellingsmodellen vanuit het perspectief van mode collapse. Hiervoor stellen we een evaluatiekader voor op basis van homotopieklassen, met de nadruk op veiligheidkritische situaties waarin de paden van verkeersdeelnemers elkaar kunnen kruisen of beïnvloeden. Binnen dit kader introduceren we maatstaven voor het kwantificeren van "mode collapse", "mode correctness" en "mode coverage". We analyseren ook hoe voorspellingen in de tijd veranderen, om gedrag te detecteren zoals het wisselen van de meest waarschijnlijke modus tussen opeenvolgende voorspellingen. We constateren dat bestaande evaluatiemaatstaven voor gezamenlijke trajectvoorspelling – zoals minADE/FDE of most likely ADE/FDE – er niet in slagen om zinvol te beoordelen of een model daadwerkelijk alle relevante modi van een gezamenlijke verdeling weet vast te leggen. Uit experimenten met verschillende modellen blijkt dat mode collapse daadwerkelijk optreed en dat sommige modellen niet in staat zijn de juiste modus te voorspellen, zelfs wanneer de interactie vrijwel onvermijdelijk wordt.

Samenvattend behandelt deze thesis het probleem van probabilistische trajectvoorspelling op meerdere niveaus. Ze verbetert de dichtheidsschatting van complexe, multimodale verdelingen, wat niet alleen relevant is voor de evaluatie van probabilistische voorspellingsmodellen, maar ook toepasbaar is op een breed scala aan problemen in diverse domeinen. Daarnaast onderzoekt zij hoe de structuur van probabilistische voorspellingsmodellen kan worden verbeterd voor zowel marginale als gezamenlijke verdelingen. Tot slot wordt een nieuw evaluatiekader voorgesteld voor de beoordeling van de voorspelde modi in gezamenlijke verdelingen. Deze bijdragen versterken het vermogen van autonome voertuigen om te redeneren over menselijke intenties en onzekerheid, wat leidt tot veiliger en beter geïnformeerde besluitvorming.

Naast het verbeteren van veiligheid en betrouwbaarheid bieden de voorgestelde benaderingen ook bredere maatschappelijke voordelen. Nauwkeurigere trajectvoorspellingen kunnen leiden tot een vlottere verkeersdoorstroming, waardoor opstoppigen en daarmee samenhangende CO_2-uitstoot worden verminderd. De ontwikkelde methodologieën sluiten bovendien aan bij smart city-initiatieven door voorspellende coördinatie van autonome voertuigen mogelijk te maken, wat datagestuurde mobiliteit en efficiënter gebruik van stedelijke vervoersnetwerken bevordert. Daarnaast kunnen deze verbeteringen, door veiliger en voorspelbaarder navigatie te ondersteunen, de toegankelijkheid vergroten voor ouderen en mensen met een beperking. Samen bevorderen deze bijdragen niet alleen het vakgebied van probabilistische trajectvoorspelling, maar ook de ontwikkeling van duurzame, inclusievere en intelligentere transportsystemen.

Acknowledgments

After four years of PhD, and six at TU Delft and in Delft, this important chapter of my life is coming to a close. While some say the PhD is an individual journey with the goal of turning you into an independent researcher, at the end of the day it is the people around you who contribute the most in how you experience this journey. This chapter goes out to all the people who accompanied me throughout this chapter of my life. Thank you all for the support and wonderful memories!

I want to start by thanking my promotors, Jens Kober and Javier Alonso-Mora. Jens, your guidance and support during my PhD were invaluable. You were always there, be it to brainstorm and discuss about ideas or provide insight on how to handle a myriad of different situations that came up throughout my PhD. You always had my back, even if things did not always go according to plan. You supported me until the very end and for this you have my deepest gratitude. Javier, even though I was a bit of a black sheep with my topic you ensured to integrate me with the AMR group. Your door was always open to talk about any matters that arose. I also want to thank Arkady Zgonnikov. Although you never officially supervised me, through our collaborations you inadvertently ended up as a mentor to me. From preparing for conferences, to discussing topics at the intersection of our fields and even involving me in projects within your group. Your positive attitude and engagement made for very enjoyable collaborations. My sincere thanks also go to my committee members Alexandre Alahi, Fernando García Fernandez, Joost de Winter, Julian Kooij, and Riender Happee for taking the time to read and evaluate this thesis.

During my time at TU Delft, I met many people who helped turn the challenges of the PhD into an enjoyable journey, especially at the beginning. First, I would like to thank Carlos. We initially met during my Master's, and while our contracts at the CoR department did not overlap we still stayed in touch for my first two years as a PhD. Even if circumstances were not always the best, you were always there to provide perspective and lend an ear. A wholehearted thank you as well to Giovanni. The projects I did under your mentorship in my last year of Master's coupled with your enthusiasm for research played a key role in me going down the route of a PhD and in us becoming colleagues. Even though our topics then diverged, I could still always count on you to have time for musing over the problematic I was tackling. And while I will probably not be moving to Abu Dhabi in the near future, I hope that my transitioning back to robot manipulation will give us new opportunities to work together again. I would also like to thank Rodrigo, whose extensive knowledge on machine learning and open-minded guidance pointed me down a promising direction when I was just starting my PhD. Normalizing Flows may not have become the new Gaussian Processes, but they proved both useful and fascinating to explore.

In my four years at CoR I moved offices several times, which, much like moving countries, allowed me to connect with many wonderful colleagues. To my office mates of F-1-460 – Alex, Italo, Heye, Hooman, Khaled (the only constant in all my offices and who I am thankful to have been able to share the craziness of our project with) Ravi,

Salvo, Stefano, and Xiangyu – thank you for the warm welcome, especially during the period of lockdowns and remote work. While the time together had been short, the bonds created stretched through my PhD. Just as I am able to make myself at home anywhere I go, I also made myself at home in an office that was never officially mine. Here I have to thank my adoptive (or adopted?) office mates at F-1-480. Thank you to Julian, my closest collaborator and good friend, Ashwin, Gustavo, Lorenzo, Irene, and Juliane for the positivity, conversations, and shared breaks that brightened many days, and to Micah for the lively banter and memorable get-togethers.

A very heartfelt thank you to my office mates in F-2-140, with whom I spent the most time and grew particularly close to. Thank you Elia for inviting us over for home-made Italian pizza; Yujie for preparing a truly lavish hot-pot experience for us all; Nils for our little chats and your recommendations on alternative rock bands; and Saray for your open and bubbly personality - I hope to one day visit you in Limburg and explore the nature in your area. Thank you Maxi for often opening your door to us and bringing us together for office events. Also, thank you for the support you provided me surrounding my considerations regarding Postdoc positions. Of course, a very big thank you to Luzia as well and all the little things we did together, from going to see a kite show, teaching me to make Bärentatzen, traveling to the ends of the Netherlands to go to a hidden magic shop, and many more. With my "jet-set" life I should really find a way to visit you in Sweden during your stay there. Lastly, Dennis; we only truly bonded in the last year or so which was possibly the most difficult period for both of us. It was always comforting to take a moment, grab a tea and have a chat before delving back into the problems we were chipping away at. To all of you, many thanks for the countless tea breaks and office events – from dinners and drinks, to skiing/snowboarding, go-karting, movies, and bouldering – which were always a welcome respite after long days of work and moments I will remember fondly for a long time to come. Last but not least, from our cozy little office at F-2-120, a sincere thank you to Ahmad, whose enthusiasm, support, and encouragement were invaluable during his time here.

I count myself lucky to have been part of such a sociable department full of wonderful people. From Sagar and his puns, Bence and our concert nights, Linda and her patience with training my voice while I was part of the 3ME choir, and Thomas my fellow "late"-luncher. And before I go into a long list of people who all contributed to a pleasant time – a big thank you to everyone else from CoR, past and present, for all the discussions and every day conversations around the department.

It goes without saying that work is only a fragment of the full story. Jointly with the start of my PhD I started actively bouldering. It is an interesting sport, not only from the physical aspect of it but also the psychological lessons it brings with it. Progress comes slowly, frustration is inevitable, and success often follows repeated failure; but with consistency, reflection, and perseverance, you keep moving forward. And while climbing can be deeply individual, the right people make the journey far more rewarding. For me those people were firstly the Delftse Spider Monkeys (Aurora, Jorge, Corrado, Patricija, Álvaro, Natalia, Elia, Daniela, and Lasse), with whom I climbed for most of my PhD. In my last year I was welcomed into another group dubbed "The Spiders" (Val, Bernat, Nikos, Sara, Wei Jun, Roy, Bart, Annabelle, and Micah), who have become rather dear to me over this relatively short period of time.

A special thank you goes to Val. You not only integrated me into the group but thought of me when putting together plans, from board-game evenings to cycling tours. You quickly became one of the people I felt closest to and could easily rely on. Even over such a short span of time, we shared a number of memorable moments – with one of the most memorable ones being the three day cycling trip together with Bernat, Markus, and Nikos – and I look forward to sharing many more!

To all my other friends who are strewn about either geographically or professionally. To Nima who I initially met at an airport in Shanghai on our way to a conference in Jeju. While oddly enough we did not interact much at the conference itself you are the only one I retained active contact with. It is in fact from your office in Utrecht that I have typed up a good portion of these acknowledgments. I appreciate you introducing me to the members of your group (research and otherwise), and it has truly been a pleasure spending time with you, Giovanni, Ira, Sylvia, Jeremy, and the others in what were probably to be my last weeks in the Netherlands. While the time was short lived, I do hope to see you all in the future, and you will all be welcome wherever I happen to settle down next.

To Enrique – our paths crossed and intertwined through a series of coincidences. You were perhaps my biggest support in the last two years of my journey despite the hundreds of kilometers which were between us for most of it, and for that I am truly grateful.

To Kaush, a steadfast friend since our Master's. Even with the distance between us, your door has always been open – in a few cases quite literally. I have always appreciated our catch-up meetups however far apart they were.

To Veronika, one of the few people with whom I could fully embrace my inner geek. I always enjoyed our long weekend meetups, whether out and about doing activities or relaxing at home over a game session.

To Dominik, with whom, despite the long periods of silence, our friendship always picks up exactly where it left off. Your regular trips back to Delft have made meeting up easy so far, and I'm sure we'll continue finding ways to reconnect. For now, at least we have our one day tour around lake Constance to look forward to!

To Ali, who by sheer coincidence remained my neighbor for most of my stay in Delft. We grew particularly close during my PhD through our frequent conversations spanning from casual banter to deep, grounding discussions, which often helped to put things into perspective.

To Jacob, one of my oldest and closest friends. Despite the distance, which characterized most of the time we have known each other, we have still managed to maintain our friendship. Unfortunately the plan to be around Frankfurt for half a year fell through, but it's my turn to visit so one way or another I will see you soon.

To Christiaan; thank you for all our coffee breaks. You were my confidant through the highs and lows of the PhD. No matter what was happening, a coffee break was never too far away. We shared in the successes and in the setbacks, and when things would start to be overwhelming our conversations would help lighten the load.

To Mariangela – when I first met you back in our Master's it was truly friendship at first sight. I was thrilled when I learned that you would be staying in the Netherlands for a PhD. Despite our crazy lives and schedules we always found a way to meet up. I have always been able to rely on you and even though we did not see each other daily or even weekly you were always there for me.

Last, but certainly not least, a very big thank you to my parents, who have always been by my side, who supported me and were with me every step of the way. Thank you for being my guides, my moral support, and the ones who always reminded me of what truly matters.

Anna
Delft, January 2026

1

Introduction

This chapter frames the problem of probabilistic trajectory prediction within the broader scope of autonomous driving in urban environments. It briefly identifies the research gaps in terms of both the development of such prediction models as well as their evaluation. The resulting contributions made to address these gaps are outlined along with the structure of this thesis.

1

1.1 Motivation

Navigating an urban environment, like a town or a city, can be surprisingly difficult. Streets are often crowded with cars, buses, cyclists, and pedestrians, all moving at different speeds and following (or in some cases even not following) their own set of rules. Construction, protests, and unexpected detours can further complicate even the simplest of routes.

While humans are generally capable of solving complex tasks and navigating in such complex environments, we are not perfect. A 2013 study across six European countries identified that human error contributes to about 80% of traffic accidents [1]. Over the years, the European Commission has been working to reduce the number of traffic accidents [2], through stricter regulations on the use of safety equipment, vehicle maintenance, and the enforcement of traffic rules. However, some contributing factors are not as easy to detect as others. Incorrect assessment of a situation [3], slow reaction time to unexpected events [4], and stress [5], are for example difficult to catch in time as an external observer, making it challenging to prevent accidents.

Another important aspect to consider about urban environments, is that not everyone can drive, be it due to age or disability. In 2023, it was estimated that about 27% of people older than 15 suffered from a disability, and this number is expected to increase further with the aging population [6–8]. Certain areas in cities, particularly those in the outskirts are often insufficiently connected through public transport, and taxi services can be costly in the long term. Having a personal means of transportation could significantly increase mobility and independence for these individuals [9, 10].

Autonomous vehicles (AVs) hold the promise of improving both road safety and accessibility. Equipped with advanced sensors, often consisting of a combination of LiDAR, radar, and cameras, AVs can monitor their surroundings, including blind spots of a vehicle for an improved overview of a situation. Advanced software is expected to enable real-time decision-making, allowing AVs to navigate the streets, without the need for human input.

However, achieving this is easier said than done. While companies like Cruise, Tesla, and Waymo have been deploying autonomous vehicles in the US in recent years, similar developments are yet to take off in Europe. This is partly due to the stricter regulations in Europe [11], but also because the infrastructure of European cities can be rather chaotic, with narrow streets, dense pedestrian and cyclist traffic, and sometimes downright confusing interchanges (see Figure 1.1 for examples) [12]. These challenges add to the difficulty of introducing AVs to European streets.

Many of the situations one will find in urban driving involve navigating around other traffic participants, ensuring both a safe traversal of the streets and avoiding to cause congestion. For an AV to achieve this, it needs to perform three tasks. First, it needs to perceive its surroundings by detecting the road structure and traffic participants around it. Secondly, it needs to anticipate, i.e. predict, the intentions of the surrounding participants. Thirdly, with the predicted intentions of the other participants, the AV then has to plan its own path to safely navigate around participants while ensuring a steady flow of traffic.

Given the dynamic and fast-paced nature of driving, particularly in urban traffic, it is crucial to anticipate the future behavior of others to ensure timely reactions without compromising on comfort and safety. Additionally, there is variability in people's actions which is in part inherent to the human decision process [15], but also, in part, a result of underlying decision factors which are unobservable by us as an external observer. The goal

Figure 1.1: Examples of challenging traffic situations in Europe. Left: narrow street in Tropea, Italy (used under license from RasaBasa - stock.adobe. [13]). Middle: dense cyclist traffic in Amsterdam, Netherlands (own work). Right: Magic Roundabout, Swindon, UK (used under license from James131/Wirestock Creators - stock.adobe.com [14]).

of this thesis is to therefore develop probabilistic trajectory prediction models, including relevant considerations that need to be made for their evaluation and their application in a downstream uncertainty-aware planner.

1.2 Research Questions

Before getting into the development of a probabilistic prediction model itself, we have to consider the means by which we evaluate it. Beyond the metrics themselves, we have to account for the fact that a vast majority of trajectory prediction model merely generate samples. This means that in order to apply probabilistic metrics such as, Brier-minFDE [16] Negative-Log-Likelihood, and most likely ADE/FDE [17], to name a few, the probability density function (PDF) first needs to be estimated based on the generated samples.

To do so, Kernel Density Estimation (KDE) is generally used [17, 18] due to its ease of use. The PDF obtained through KDE is greatly dependent on the choice of kernel [19]. Particularly in the context of multi-modal distributions, it is well know that the commonly used Silverman's rule of thumb, leads to over-smoothing in the case of multi-modal distributions [20].

In order to have a fair comparison between different models, it is therefore important to have an accurate estimation of the underlying PDF. Additionally, since it would be difficult and time consuming to manually analyze the evaluation data in trajectory prediction datasets, we would need a non-parametric approach to estimate the PDF. This brings us to our first research question.

Question 1 *How can we ensure good probability density estimation irrespective of the type of distribution?*

Now that we have a way to evaluate our trajectory prediction models, we can focus on developing the models themselves. Keeping in mind the overarching goal of a safe yet efficient AV, we have to consider what the most beneficial output format would be. While predictions in the form of Gaussian distributions at each time steps are commonly used in uncertainty-aware planning [21] due to their simple mathematical formulation and the computational ease that follows from it, they also lead to rather conservative plans [22].

For this reason, it is beneficial to have a prediction model, which can provide distributions over the complete trajectories along with their corresponding likelihoods. This raises the following research question;

Question 2 *How can we leverage machine learning models to learn explicit distributions over trajectories?*

Nevertheless, it is generally not enough to predict the intentions of every agent in a scene independently. Imagine the case of two cars wanting to cross an intersection. While we can provide the past trajectories of both agents as part of the input to the model, a marginal prediction model does not consider how the interaction of these agents will evolve. As such, the most likely marginal predictions for the two agents can easily end up being that they both go straight, thus placing a high probability on the two agents colliding. In reality, people will generally aim to mitigate collisions, meaning it is more likely for one agent to give way to the other agent. For this reason, it is beneficial to learn a joint distribution (capturing the future evolution of all agents in a scene simultaneously) instead of relying purely on marginal distributions (predicting all agent independently).

Since the amount of available data is fairly small compared to the dimensionality and complexity of the joint distributions, it can be beneficial to impose a structure within the model which factorizes the joint distribution into smaller conditional distributions. Intuitively, this factorization would aim to capture how the future behavior of certain agents will affect the future behavior of those around them. However, to establish this type of factorization, it is necessary to have a means for representing which agent influences which other agents. This gives us our third research question:

Question 3 *How should we model the interaction between agents and leverage this for predicting joint distributions?*

Lastly, before being able to deploy a learned model on a real system such as an AV, it is important to evaluate them. However, when dealing with multi-modal distributions over complex data, it is challenging to evaluate their quality. Established metrics, such as minimum average and final displacement error (minADE and minFDE), negative log likelihood of the ground truth w.r.t. the predicted distribution, and Brier minADE, often fail to provide sufficient insights. One of the things which makes this challenging is that for any given input consisting of different configurations of the agents and environment there is only ever one corresponding ground truth. This is particularly relevant in the case of potential future interactions between traffic participants where the paths of two or more agents may cross. In this case, if a model completely disregards the mode of an agent cutting in in-front of the AV, this could lead to a collision that could have otherwise been avoided with a bit of added foresight from the predictions. This problematic raises our final research question:

Question 4 *How can we evaluate the predicted modes for multi-modal joint distributions?*

1.3 Contributions and Thesis Outline

This thesis advances the field of probabilistic trajectory prediction through contributions spanning both the modeling and evaluation of uncertainty in human behavior in urban environments. The work address key challenges related to density estimation, multi-modality, and the reliable assessment of probabilistic prediction models in complex, interactive driving scenarios. The following chapters of this thesis each address one of the research questions in the order posed above. An overview of their main topic and corresponding contributions is provided below:

[Chapter 2] A Robust Multi-Modal Density Estimator: We propose a density estimation method which provides improved density estimations over high-dimensional, multi-modal, and non-Gaussian distributions compared to the state of the art. The proposed density estimator is used throughout this thesis as the underlying estimator over samples generated by different prediction models. The probability density function over these samples is then used to calculate and compare the performance metrics between models to ensure a fair comparison.

[Chapter 3] A Probabilistic Trajectory Prediction Model: We introduce TrajFlow, a probabilistic trajectory prediction model that leverages Normalizing Flows and incorporates an RNN-AE to learn distributions over compact trajectory encodings. This approach simplifies the learning process, captures complex multi-modal behavior in human motion, and demonstrates improved performance over state-of-the-art trajectory prediction methods.

[Chapter 4] A Study on Different Approaches for Representing the Influence Between Agents: We investigate the influence of different ways to represent the interactions between agents within the scope of joint trajectory prediction. The investigation reveals that neural networks on their own struggle to learn the interactive connections between agents, and that prediction models can benefit from the integration of more explicit interaction representations.

[Chapter 5] A Framework for Evaluating Critical Interactions in Joint Trajectory Prediction Models: We proposed a novel evaluation framework for joint trajectory prediction based on homotopy classes. Within this framework we introduced metrics for mode collapse, mode correctness, and coverage with a focus on safety-critical interactions. The proposed framework enables a more holistic assessment of prediction models and provides AV developers with tools to identify and address critical weaknesses that could otherwise compromise safe motion planning.

This thesis is concluded in Chapter 6, with a discussion of the findings and the future outlooks.

2

ROME: Robust Multi-Modal Density Estimator

The estimation of probability density functions is a fundamental problem in science and engineering. However, common methods such as kernel density estimation have been shown to lack robustness, while more complex methods have not been evaluated in multi-modal estimation problems. One such multi-modal estimation problem arises during the evaluation of probabilistic trajectory prediction models. A majority of prediction models generate samples instead of analytical formulations of the distribution they are estimating. As such, density estimation needs to be applied onto the generated samples in order to then calculate relevant metrics such as Estimated Calibration Error, Jensen-Shannon Divergence, and Negative Log-Likelihood to name a few. To ensure an unbiased evaluation, it is important that the density estimation is as accurate as possible. This chapter introduces ROME (RObust Multi-modal Estimator), a non-parametric approach for density estimation which addresses the challenge of estimating multi-modal, non-normal, and highly correlated distributions. ROME utilizes clustering to segment a multi-modal set of samples into multiple uni-modal ones and then combines simple KDE estimates obtained for individual clusters in a single multi-modal estimate. We compared our approach to state-of-the-art methods for density estimation as well as ablations of ROME, showing that it not only outperforms established methods but is also more robust to a variety of distributions. Our results demonstrate that ROME can overcome the issues of over-fitting and over-smoothing exhibited by other estimators.

This chapter is a verbatim copy of the peer-reviewed paper [23]: 🖹 ***A. Mészáros***, J. F. Schumann*, J. Alonso-Mora, A. Zgonnikov‡, and J. Kober‡. "ROME: Robust multi-modal density estimator." In Proceedings 33rd International Joint Conference on Artificial Intelligence 2024.* * Indicates joint first-author contribution. ‡ Indicates joint last-author contribution.

Statement of contributions: Anna Mészáros and Julian F. Schumann equally contributed to the design and implementation of the research, to the analysis of the results and to the writing of the manuscript and should be considered joint first authors. Javier Alonso-Mora provided valuable feedback on the writing of the manuscript. Arkady Zgonnikov and Jens Kober provided valuable feedback at all steps of the project and should be considered joint last authors.

2.1 Introduction

Numerous processes are non-deterministic by nature, from geological and meteorological occurrences, biological activities, as well as the behavior of living beings. Estimating the underlying probability density functions (PDFs) of such processes enables a better understanding of them and opens possibilities for probabilistic inference regarding future developments. Density estimation is instrumental in many applications including classification, anomaly detection, and evaluating probabilistic AI models, such as generative adversarial networks [24], variational autoencoders [25], normalizing flows [26], and their numerous variations.

When probabilistic models are trained on multi-modal data, they are often evaluated using simplistic metrics (e.g., mean squared error (MSE) between the predicted and ground truth samples). However, such simplistic metrics are unsuited for determining how well a predicted distribution corresponds to the underlying distribution, as they do not capture the fit to the whole distribution. For example, the lowest MSE value between true and predicted samples could be achieved by accurate predictions of the mean of the true underlying distribution whereas potential differences in variance or shape of the distribution would not be penalized. This necessitates more advanced metrics that evaluate the match between the model and the (potentially multi-modal) data. For instance, negative log-likelihood (NLL), Jensen-Shannon divergence (JSD), and expected calibration error (ECE) can be used to evaluate how well the *full distribution* of the data is captured by the learned models [27–29]. However, most data-driven models represent the learned distribution implicitly, only providing individual samples and not the full distribution as an output. This complicates the comparison of the model output to the ground-truth data distributions since the above metrics require distributions, not samples, as an input. Practically, this can be addressed by estimating the predicted probability density based on samples generated by the model.

Simple methods like Gaussian mixture models (GMM), kernel density estimation (KDE) [30], and k-nearest neighbors (kNN) [31] are commonly used for estimating probability density functions. These estimators however rely on strong assumptions about the underlying distribution, and can thereby introduce biases or inefficiencies in the estimation process. For example, problems can arise when encountering multi-modal, non-normal, and highly correlated distributions (see Section 2.2). While more advanced methods such as manifold Parzen windows (MPW) [32] and vine copulas (VC) [33] exist, they have not been thoroughly tested on such problems, which raises questions about their performance.

To overcome these limitations, we propose a novel density estimation approach: RObust Multi-modal Estimator (ROME). ROME employs a non-parametric clustering approach to segment potentially multi-modal distributions into separate uni-modal ones (Figure 2.1). These uni-modal sub-distributions are then estimated using a downstream probability density estimator (such as KDE). We test our proposed approach against a number of existing density estimators in three simple two-dimensional benchmarks designed to evaluate a model's ability to successfully reproduce multi-modal, highly-correlated, and non-normal distributions. Finally, we test our approach in a real-world setting using a distribution over future trajectories of human pedestrians created based on the Forking Paths dataset [34].

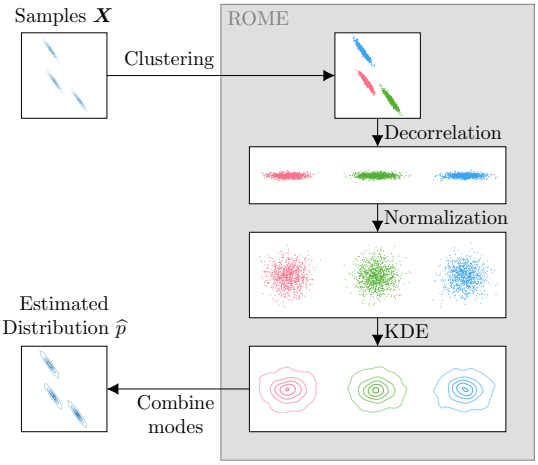


Figure 2.1: ROME takes samples from unknown distributions and estimates their densities to enable further downstream applications.

2.2 Related Work

The most common class of density estimators are so-called Parzen windows [35], which estimate the density through an aggregation of parametric probability density functions. A number of common methods use this approach, with KDE being a common non-parametric method [36]. Provided a type of kernel – which is oftentimes a Gaussian but can be any other type of distribution – KDE places the kernels around each sample of a data distribution and then sums over these kernels to get the final density estimation over the data. This method is often chosen as it does not assume any underlying distribution type [36]. However, if the underlying distribution is highly correlated, then the common use of a single isotropic kernel function can lead to over-smoothing [32, 37]. Among multiple approaches for overcoming this issue [37, 38], especially noteworthy is the MPW approach [32]. It uses a unique anisotropic kernel for every datapoint, estimated based on the correlation of the k-nearest neighbors of each sample. However, it has not been previously tested in high-dimensional benchmarks, which is especially problematic as the required memory scales quadratically with the dimensionality of the problem.

Another common subtype of Parzen windows are GMMs [30], which assume that the data distribution can be captured through a weighted sum of Gaussian distributions. The parameters of the Gaussian distributions – also referred to as components – are estimated through expected likelihood maximization. Nonetheless, especially for non-normal distributions, one needs to have prior knowledge of the expected number of components to achieve a good fit without over-fitting to the training data or over-smoothing [39].

Besides different types of Parzen windows, a prominent alternative is kNN [31] which uses the local density of the k nearest neighbors of every data point to estimate the overall

density. While this method is non-parametric, it cannot be guaranteed that the resulting distribution will be normalized [36]. This could be rectified by using Monte Carlo sampling to obtain a good coverage of the function's domain and obtain an accurate estimate of the normalization factor, which, however, becomes computationally intractable for high-dimensional distributions.

When it comes to estimating densities characterized by correlations between dimensions, copula-based methods are an often favored approach. Copula-based methods decompose a distribution into its marginals and an additional function, called a copula, which describes the dependence between these marginals over a marginally uniformly distributed space. The downside of most copula-based approaches is that they rely on choosing an adequate copula function (e.g., Gaussian, Student, or Gumbel) and estimating their respective parameters [40]. One non-parametric copula-based density estimator [41] aims to address this limitation by estimating copulas with the help of superimposed Legendre polynomials. While this can achieve good results in estimating the density function, it may become computationally intractable as the distribution's dimensionality increases. Another approach involves the use of VC [33], which assume that the whole distribution can be described as a product of bivariate Gaussian distributions, thus alleviating the curse of dimensionality. Its convergence, however, can only be guaranteed for normal distributions. Elsewhere [42], a similar approach was pursued, with changes such as using logarithmic splines instead of univariate KDEs for estimating the marginals. However, both of these approaches are not designed for multi-modal distributions and have not been thoroughly tested on such problems.

2.3 RObust Multi-modal Estimator (ROME)

The problem of density estimation can be formalized as finding a queryable $\widehat{p} \in \mathcal{P}$, where

$$\mathcal{P} = \left\{ g : \mathbb{R}^M \to \mathbb{R}^+ \,|\, \int g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 1 \right\},$$

such that $\widehat{p}$ is close to the non-queryable PDF p underlying the N available M-dimensional samples $\boldsymbol{X} \in \mathbb{R}^{N\times M}$: $\boldsymbol{X} \sim p$.

A solution to the above problem would be an estimator $f : \mathbb{R}^{N\times M} \to \mathcal{P}$, resulting in $\widehat{p} = f(\boldsymbol{X})$. Our proposed estimator f_{ROME}[1] (Algorithm 1) is built on top of non-parametric cluster extraction. Namely, by separating groups of samples surrounded by areas of low density – expressing the mode of the underlying distribution – we reduce the multi-modal density estimation problem to multiple uni-modal density estimation problems for each cluster. The distributions within each cluster then become less varied in density or correlation than the full distribution. Combining this with decorrelation and normalization, the use of established methods such as KDE to estimate probability densities for those uni-modal distributions is now more promising, as problems with multi-modality and correlated modes (see Section 2.2) are accounted for. The multi-modal distribution is then obtained as a weighted average of the estimated uni-modal distributions.

[1] Source code: `https://github.com/anna-meszaros/ROME`

Algorithm 1 ROME

function TRAINROME($\boldsymbol{X}$)
  ▷ *Clustering (OPTICS)*
  $\boldsymbol{X}_{I,N}, \boldsymbol{R}_N \leftarrow$ REACHABILTYANALYSIS($\boldsymbol{X}$)
  $\boldsymbol{C}, S \leftarrow \{\{1,\dots,N\}\}, -1.1$
  for all $\epsilon \in \boldsymbol{\varepsilon}$ **do**
    $\boldsymbol{C}_\epsilon \leftarrow$ **DBSCAN**($\boldsymbol{R}_{I,N}, \epsilon$)
    $S_\epsilon \leftarrow$ **SIL**($\boldsymbol{C}_\epsilon, \boldsymbol{X}_{I,N}$)
    if $S_\epsilon > S$ **then**
      $\boldsymbol{C}, S \leftarrow \boldsymbol{C}_\epsilon, S_\epsilon$
  for all $\xi \in \boldsymbol{\xi}$ **do**
    $\boldsymbol{C}_\xi \leftarrow \boldsymbol{\xi}$**-clustering**($\boldsymbol{R}_{I,N}, \xi$)
    $S_\xi \leftarrow$ **SIL**($\boldsymbol{C}_\xi, \boldsymbol{X}_{I,N}$)
    if $S_\xi > S$ **then**
      $\boldsymbol{C}, S \leftarrow \boldsymbol{C}_\xi, S_\xi$
  for all $C \in \boldsymbol{C}$ **do**
    ▷ *Decorrelation*
    $\overline{\boldsymbol{x}}_C \leftarrow \text{MEAN}(\boldsymbol{X}_C)$
    $\overline{\boldsymbol{X}}_C \leftarrow \boldsymbol{X}_C - \overline{\boldsymbol{x}}_C$
    $\boldsymbol{R}_C \leftarrow PCA(\overline{\boldsymbol{X}}_C)$
    ▷ *Normalization*
    $\widetilde{\boldsymbol{\Sigma}}_C \leftarrow \text{STD}(\overline{\boldsymbol{X}}_C \boldsymbol{R}_C^T)$
    $\boldsymbol{T}_C \leftarrow \boldsymbol{R}_C^T \widetilde{\boldsymbol{\Sigma}}_C^{-1}$
    ▷ *PDF Estimation*
    $\widehat{p}_C \leftarrow f_{\text{KDE}}\left(\overline{\boldsymbol{X}}_C \boldsymbol{T}_C\right)$
  return $\boldsymbol{C}, \left\{\widehat{p}_C, \overline{\boldsymbol{x}}_C, \boldsymbol{T}_C \,|\, C \in \boldsymbol{C}\right\}$

function QUERYROME($\boldsymbol{x}$, $\boldsymbol{X}$)
  $\boldsymbol{C}, \left\{\widehat{p}_C, \overline{\boldsymbol{x}}_C, \boldsymbol{T}_C \,|\, C \in \boldsymbol{C}\right\} \leftarrow$ TRAINROME($\boldsymbol{X}$)
  $l = \boldsymbol{0}$
  for all $C \in \boldsymbol{C}$ **do**
    $\widehat{\boldsymbol{x}} \leftarrow (\boldsymbol{x} - \overline{\boldsymbol{x}}_C)\, \boldsymbol{T}_C$
    $l \leftarrow l + \ln(\widehat{p}_C(\widehat{\boldsymbol{x}})) + \ln(|C|) - \ln N + \ln(|\det(\boldsymbol{T}_C)|)$
  return $\exp(l)$

2.3.1 EXTRACTING CLUSTERS

To cluster samples $\boldsymbol{X}$, ROME employs the OPTICS algorithm [43] that can detect clusters of any shape with varying density using a combination of reachability analysis – which orders the data in accordance to reachability distances – followed by clustering the ordered data based on these reachability distances.

In the first part of the algorithm, the reachability analysis is used to sequentially transfer samples from a set of unincluded samples $\boldsymbol{X}_{U,i}$ to the set of included and ordered samples $\boldsymbol{X}_{I,i}$, starting with a random sample $\boldsymbol{x}_1$ ($\boldsymbol{X}_{I,1} = \{\boldsymbol{x}_1\}$ and $\boldsymbol{X}_{U,1} = \boldsymbol{X} \backslash \{\boldsymbol{x}_1\}$). The samples $\boldsymbol{x}_{i+1}$

are then selected at iteration i based on the reachability distance r:

$$\boldsymbol{x}_{i+1} = \underset{\boldsymbol{x}\in\boldsymbol{X}_{U,i}}{\arg\min}\, r(\boldsymbol{x}, \boldsymbol{X}_{I,i}) = \underset{\boldsymbol{x}\in\boldsymbol{X}_{U,i}}{\arg\min}\ \min_{\widetilde{\boldsymbol{x}}\in\boldsymbol{X}_{I,i}}\ d_r(\boldsymbol{x}, \widetilde{\boldsymbol{x}})\,. \tag{2.1}$$

This sample is then transferred between sets, with $\boldsymbol{X}_{I,i+1} = \boldsymbol{X}_{I,i} \cup \{\boldsymbol{x}_{i+1}\}$ and $\boldsymbol{X}_{U,i+1} = \boldsymbol{X}_{U,i} \backslash \{\boldsymbol{x}_{i+1}\}$, while expanding the reachability set $\boldsymbol{R}_{i+1} = \boldsymbol{R}_i \cup \left\{ r(\boldsymbol{x}_{i+1}, \boldsymbol{X}_{I,i}) \right\}$ (with $\boldsymbol{R}_1 = \{\infty\}$). The reachability distance d_r in Equation (2.1) is defined as

$$d_r(\boldsymbol{x}, \widetilde{\boldsymbol{x}}) = \max\left\{ \|\boldsymbol{x} - \widetilde{\boldsymbol{x}}\|,\ \min_{k_{\boldsymbol{c}}}_{\widehat{\boldsymbol{x}}\in\boldsymbol{X}\backslash\{\boldsymbol{x}\}} \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\| \right\},$$

where $\min_{k_{\boldsymbol{c}}}$ is the $k_{\boldsymbol{c}}$-smallest value of all the available $\widehat{\boldsymbol{x}}$, used to smooth out random local density fluctuations. We use

$$k_{\boldsymbol{c}} = \min\left\{ k_{\max}, \max\left\{ k_{\min}, \frac{NM}{\alpha_k} \right\} \right\}, \tag{2.2}$$

where $k_{\min}$ and $k_{\max}$ ensure there are sufficient but not too many points for this smoothing, while the term NM/α_k adjusts $k_{\boldsymbol{c}}$ to the number of samples and dimensions.

The second part of the OPTICS algorithm – after obtaining the reachability distances $\boldsymbol{R}_N$ and the ordered set $\boldsymbol{X}_{I,N}$ – is the extraction of a set of clusters $\boldsymbol{C}$, with clusters $C = \{c_{\min}, \dots, c_{\max}\} \in \boldsymbol{C}$ (with $\boldsymbol{X}_C = \{\boldsymbol{x}_{I,N,j} \,|\, j \in C\} \in \mathbb{R}^{|C|\times M}$). As the computational cost of creating such a cluster set is negligible compared to the reachability analysis, we can test multiple clusterings generated using two different approaches (with $r_{\text{bound}} = \min\{r_{N,c_{\min}}, r_{N,c_{\max}+1}\}$, see Appendix A.1 for further discussion):

- First, we use **DBSCAN** [44] for generating the clustering $\boldsymbol{C}_\epsilon$ based on an absolute limit ϵ, where a cluster must fulfill the condition:

$$r_{N,c} < \epsilon \leq r_{\text{bound}}\ \forall\, c \in C\backslash\{c_{\min}\}. \tag{2.3}$$

- Second, we use **ξ-clustering** [43] to generate the clustering $\boldsymbol{C}_\xi$ based on the proportional limit ξ, where a cluster fulfills:

$$\xi \leq 1 - \frac{r_{N,c}}{r_{\text{bound}}}\ \forall\, c \in C\backslash\{c_{\min}\}. \tag{2.4}$$

In both cases, each prospective cluster C also has to fulfill the condition $|C| \geq 2$. However, it is possible that not every sample can be fit into a cluster fulfilling the conditions above. These samples are then kept in a separate noise cluster C_{noise} that does not have to fulfill those conditions ($C_{\text{noise}} \in \boldsymbol{C}_\epsilon$ or $C_{\text{noise}} \in \boldsymbol{C}_\xi$ respectively). Upon generating multiple sets of clusters $\boldsymbol{C}_\epsilon$ ($\epsilon \in \boldsymbol{\epsilon}$) and $\boldsymbol{C}_\xi$ ($\xi \in \boldsymbol{\xi}$), we select the final set of clusters $\boldsymbol{C}$ that achieves the highest silhouette score[2] $S = \mathbf{SIL}(\boldsymbol{C}, \boldsymbol{X}_{I,N}) \in [-1, 1]$ [45]. The clustering then allows us to use PDF estimation methods on uni-modal distributions.

[2]The silhouette score measures the similarity of each object to its own cluster's objects compared to the other clusters' objects.

2.3.2 Feature Decorrelation

In much of real-life data, such as the distributions of a person's movement trajectories, certain dimensions of the data are likely to be highly correlated. Therefore, the features in each cluster $C \in \boldsymbol{C}$ should be decorrelated using a rotation matrix $\boldsymbol{R}_C \in \mathbb{R}^{M\times M}$. In ROME, $\boldsymbol{R}_C$ is found using principal component analysis (PCA) [46] on the cluster's centered samples $\overline{\boldsymbol{X}}_C = \boldsymbol{X}_C - \overline{\boldsymbol{x}}_C$ ($\overline{\boldsymbol{x}}_C$ is the cluster's mean value). An exception are the noise samples in C_{noise}, which are not decorrelated (i.e., $\boldsymbol{R}_{C_{\text{noise}}} = \boldsymbol{I}$). One can then get the decorrelated samples $\boldsymbol{X}_{\text{PCA},C}$:

$$\boldsymbol{X}^T_{\text{PCA},C} = \boldsymbol{R}_C \overline{\boldsymbol{X}}^T_C .$$

2.3.3 Normalization

After decorrelation, we use the matrix $\widetilde{\Sigma}_C \in \mathbb{R}^{M\times M}$ to normalize $\boldsymbol{X}_{\text{PCA},C}$:

$$\widehat{\boldsymbol{X}}_C = \boldsymbol{X}_{\text{PCA},C} \left(\widetilde{\Sigma}_C\right)^{-1} = \overline{\boldsymbol{X}}_C \boldsymbol{R}_C^T \left(\widetilde{\Sigma}_C\right)^{-1} = \overline{\boldsymbol{X}}_C \boldsymbol{T}_C .$$

Here, $\widetilde{\Sigma}_C$ is a diagonal matrix with the entries $\widetilde{\sigma}_C^{(m)}$. To avoid over-fitting to highly correlated distributions, we introduce a regularization with a value $\sigma_{\min}$ (similar to [32]) that is applied to the empirical standard deviation $\sigma^{(m)}_{\text{PCA},C} = \sqrt{\mathbb{V}_m\left(\boldsymbol{X}_{\text{PCA},C}\right)}$ ($\mathbb{V}_m$ is the variance along feature m) of each rotated feature m (with $\boldsymbol{C}' = \boldsymbol{C}\backslash\{C_{\text{noise}}\}$):

$$\widetilde{\sigma}_C^{(m)} = \begin{cases} \max\left\{ \sum\limits_{\mathfrak{C}\in\boldsymbol{C}'} \frac{\sqrt{\mathbb{V}_m(\boldsymbol{X}_{\mathfrak{C}})}}{|C|-1}, \sigma_{\min} \right\} & C = C_{\text{noise}} \\ \left(1 - \frac{\sigma_{\min}}{\max\limits_m \sigma^{(m)}_{\text{PCA},C}}\right)\sigma^{(m)}_{\text{PCA},C} + \sigma_{\min} & \text{otherwise} \end{cases} .$$

2.3.4 Estimating the Probability Density Function

Taking the transformed data (decorrelated and normalized), ROME fits the Gaussian KDE f_{KDE} on each separate cluster C as well as the noise samples C_{noise}. For a given bandwidth b_C for data in cluster C, this results in a partial PDF $\widehat{p}_C$.

$$\widehat{p}_C\left(\widehat{\boldsymbol{x}}\right) = f_{\text{KDE}}\left(\widehat{\boldsymbol{X}}_C\right)\left(\widehat{\boldsymbol{x}}\right) = \frac{1}{|C|}\sum_{\boldsymbol{\chi}\in\widehat{\boldsymbol{X}}_C} \mathcal{N}\left(\widehat{\boldsymbol{x}}|\boldsymbol{\chi}, b_C\boldsymbol{I}\right) .$$

The bandwidth b_C is set using Silverman's rule [36]:

$$b_C = \left(\frac{M+2}{4} n_C\right)^{-\frac{1}{M+4}}, n_C = \begin{cases} 1 & C = C_{\text{noise}} \\ |C| & \text{else} \end{cases} .$$

To evaluate the density function $\widehat{p} = f_{\text{ROME}}(\boldsymbol{X})$ for a given sample $\boldsymbol{x}$, we take the weighted averages of each cluster's $\widehat{p}_C$:

$$\widehat{p}(\boldsymbol{x}) = \sum_{C\in\boldsymbol{C}} \frac{|C|}{N} \widehat{p}_C\left((\boldsymbol{x} - \overline{\boldsymbol{x}}_C)\boldsymbol{T}_C\right)|\det(\boldsymbol{T}_C)| .$$

Here, the term $|C|/N$ is used to weigh the different distributions of each cluster with the size of each cluster, so that each sample is represented equally. As the different KDEs $\widehat{p}_C$ are fitted to the transformed samples, we apply them not to the original sample $\boldsymbol{x}$, but instead apply the identical transformation used to generate those transformed samples, using $\widehat{p}_C((\boldsymbol{x}-\overline{\boldsymbol{x}}_C))$. To account for the change in density within a cluster C introduced by the transformation $\boldsymbol{T}_C$, we use the factor $|\det(\boldsymbol{T}_C)|$.

2.4 Experiments

We compare our approach against two baselines from the literature (VC [33] and MPW [32]) in four scenarios, using three metrics. Additionally, we carry out an ablation study on our proposed method ROME.

For the hyperparameters pertaining to the clustering within ROME (see Section 2.3.1), we found empirically that stable results can be obtained using 199 possible clusterings, 100 for DBSCAN (Equation (2.3))

$$\boldsymbol{\varepsilon} = \left\{ \min R_N + \left(\frac{\alpha}{99}\right)^2 (\max(R_N \backslash \{\infty\}) - \min R_N) \mid \alpha \in \{0,\dots,99\} \right\}$$

combined with 99 for ξ-clustering (Equation (2.4))

$$\boldsymbol{\xi} = \left\{ \frac{\beta}{100} \mid \beta \in \{1,\dots,99\} \right\},$$

as well as using $k_{\min} = 5$, $k_{\max} = 20$, and $\alpha_k = 400$ for calculating $k_{\boldsymbol{c}}$ (Equation (2.2) and Appendix A.3).

2.4.1 Distributions

In order to evaluate different aspects of a density estimation method f, we used a number of different distributions.

- Three two-dimensional synthetic distributions (Figure 2.2) were used to test the estimation of distributions with multiple clusters, which might be highly correlated (Aniso) or of varying densities (Varied), or express non-normal distributions (Two Moons).
- A multivariate, 24-dimensional, and highly correlated distribution generated from a subset of the Forking Paths dataset [34] (Figure 2.3). The 24 dimensions correspond to the x and y positions of a human pedestrian across 12 timesteps. Based on 6 original trajectories ($\boldsymbol{x}_i^* \in \mathbb{R}^{12\times 2}$), we defined the underlying distribution p in such a way, that one could calculate a sample $\boldsymbol{x} \sim p$ with:

  $$\boldsymbol{x} = s\boldsymbol{x}_i^* \boldsymbol{R}_\theta^T + \boldsymbol{L}\boldsymbol{n}, \text{ with } i \sim \mathcal{U}\{1,6\}.$$

  Here, $\boldsymbol{R}_\theta \in \mathbb{R}^{2\times 2}$ is a rotation matrix rotating $\boldsymbol{x}_i^*$ by $\theta \sim \mathcal{N}(0, \frac{\pi}{180})$, while $s \sim \mathcal{N}(1, 0.03)$ is a scaling factor. $\boldsymbol{n} = \mathcal{N}(\boldsymbol{0}, 0.03\boldsymbol{I}) \in \mathbb{R}^{12\times 2}$ is additional noise added on all dimensions using $\boldsymbol{L} \in \mathbb{R}^{12\times 12}$, a lower triangular matrix that only contains ones.

Further tests on uni-modal problems can be found in Appendix A.5.

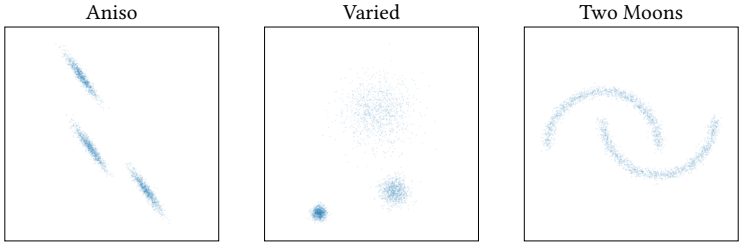


Figure 2.2: Samples from the two-dimensional synthetic distributions used for evaluating different PDF estimators.

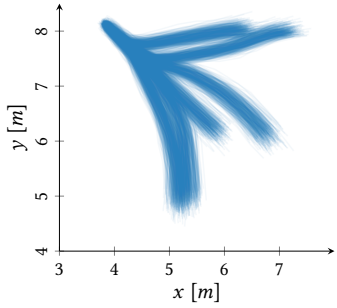


Figure 2.3: Samples from the multi-modal pedestrian trajectory distribution [34] used for evaluating different PDF estimators. Trajectories span 12 timesteps recorded at 2.5 Hz.

2.4.2 Evaluation and Metrics

When estimating density $\widehat{p}$, since we cannot query the distribution p underlying the samples $\boldsymbol{X}$, we require metrics that can provide insights purely based on those samples. To this end we use the following three metrics to quantify how well a given density estimator f can avoid both over-fitting and over-smoothing.

- To test for over-fitting, we first sample two different datasets $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ (N samples each) from p ($\boldsymbol{X}_1, \boldsymbol{X}_2 \sim p$). We then use the estimator f to create two queryable distributions $\widehat{p}_1 = f(\boldsymbol{X}_1)$ and $\widehat{p}_2 = f(\boldsymbol{X}_2)$. If those distributions $\widehat{p}_1$ and $\widehat{p}_2$ are similar, it would mean the tested estimator does not over-fit; we measure this similarity using the Jensen-Shannon divergence [47]:

$$D_{\mathrm{JS}}(\widehat{p}_1 \| \widehat{p}_2) = \frac{1}{2N \ln(2)} \sum_{\boldsymbol{x} \in \boldsymbol{X}_1 \cup \boldsymbol{X}_2} h_1(\boldsymbol{x}) + h_2(\boldsymbol{x})$$

$$h_i(\boldsymbol{x}) = \frac{\widehat{p}_i(\boldsymbol{x})}{\widehat{p}_1(\boldsymbol{x}) + \widehat{p}_2(\boldsymbol{x})} \ln\left(\frac{2\widehat{p}_i(\boldsymbol{x})}{\widehat{p}_1(\boldsymbol{x}) + \widehat{p}_2(\boldsymbol{x})} \right)$$

  This metric, however, is not able to account for systematic biases that could be

present in the estimator f. Comparisons of $\widehat{p}_1$ with p – if the dataset allows – can be found in Appendix A.4.

- To test the goodness-of-fit of the estimated density, we first generate a third set of samples $\widehat{\boldsymbol{X}}_1 \sim \widehat{p}_1$ with N samples. We then use the Wasserstein distance W [48] on the data to calculate the indicator $\widehat{W}$:

$$\widehat{W} = \frac{W(\boldsymbol{X}_1, \widehat{\boldsymbol{X}}_1) - W(\boldsymbol{X}_1, \boldsymbol{X}_2)}{W(\boldsymbol{X}_1, \boldsymbol{X}_2)}$$

  Here, $\widehat{W} > 0$ indicates over-smoothing or misplaced modes, while $-1 \leq \widehat{W} < 0$ indicates over-fitting.

- Not every density estimator f has the ability to generate the samples $\widehat{\boldsymbol{X}}_1$. Consequently, we need to test for goodness-of-fit without relying on $\widehat{\boldsymbol{X}}_1$. Therefore, we use the average log-likelihood

$$\widehat{L} = \frac{1}{N} \sum_{\boldsymbol{x} \in \boldsymbol{X}_2} \ln\left(\widehat{p}_1(\boldsymbol{x})\right),$$

  which would be maximized only for $p = \widehat{p}_1$ as long as $\boldsymbol{X}_2$ is truly representative of p (see Gibbs' inequality [49]). However, using this metric might be meaningless if $\widehat{p}_1$ is not normalized, as the presence of the unknown scaling factor makes the $\widehat{L}$ values of two estimators incomparable, and cannot discriminate between over-fitting and over-smoothing.

For each candidate method f, we used $N = 3000$, and every metric calculation was repeated 100 times to take into account the inherent randomness of sampling from the distributions, with the standard deviation in the tables being reported with $\pm$.

2.4.3 Ablations

To better understand the performance of our approach, we investigated variations in four key aspects of ROME:

- *Clustering approach.* First, we replaced the silhouette score (see Section 2.3.1) with density based cluster validation (DBCV) [50] when selecting the optimal clustering out of the 199 possibilities. Furthermore, we investigated the approach of no clustering ($\boldsymbol{C} = \{\{1, \ldots, N\}\}$) .
- *Decorrelation vs No decorrelation.* We investigated the effect of removing rotation by setting $\boldsymbol{R}_C = \boldsymbol{I}$.
- *Normalization vs No normalization.* We studied the sensitivity of our approach to normalization by setting $\widetilde{\Sigma}_C = \boldsymbol{I}$.
- *Downstream density estimator*. We replaced f_{KDE} with two other candidate methods. First, we used a single-component Gaussian mixture model f_{GMM}

$$f_{\text{GMM}}(\boldsymbol{X})(\boldsymbol{x}) = \mathcal{N}\left(\boldsymbol{x} | \widehat{\boldsymbol{\mu}}_{\boldsymbol{X}}, \widehat{\Sigma}_{\boldsymbol{X}}\right)$$

fitted to the observed mean $\hat{\boldsymbol{\mu}}$ and covariance matrix $\hat{\Sigma}$ of a dataset $\boldsymbol{X}$. Second, we used a k-nearest neighbor approach f_{kNN} [31]

$$f_{\text{kNN}}(\boldsymbol{X})(\boldsymbol{x}) = \frac{k}{N\, \mathcal{V}_M \min_{\hat{\boldsymbol{x}} \in \boldsymbol{X}} \left\| \boldsymbol{x} - \hat{\boldsymbol{x}} \right\|^M}$$

where $\mathcal{V}_M$ is the volume of the M-dimensional unit hypersphere. We used the rule-of-thumb $k = \lfloor \sqrt{N} \rfloor$. However, this estimator cannot generate samples.

While those four factors would theoretically lead to 24 estimators, f_{KDE} as well as f_{kNN} being invariant against rotation and f_{GMM} being invariant against any linear transformation means that only 14 of ROME's ablations are actually unique.

2.5 Results

2.5.1 Baseline Comparison

We found that ROME avoids the major pitfalls displayed by the two baseline methods on the four tested distributions (Table 2.1). Out of the two baseline methods, the manifold Parzen windows (MPW) approach has a stronger tendency to over-fit in the case of the two-dimensional distributions compared to ROME, as quantified by lower D_{JS} values achieved by ROME. MPW does, however, achieve a better log-likelihood for the Two Moons distribution compared to ROME. This could be due to the locally adaptive non-Gaussian distributions being less susceptible to over-smoothing than our approach of using a single isotropic kernel for each cluster if such clusters are highly non-normal. Lastly, in the case of the pedestrian trajectories distribution, ROME once more achieves better performance than MPW, with MPW performing worse both in terms of D_{JS} and $\widehat{L}$.

Meanwhile, the vine copulas (VC) approach tends to over-smooth the estimated densities (large positive $\widehat{W}$ values), and even struggles with capturing the different modes (see Figure 2.4). This is likely because VC uses KDE with Silverman's rule of thumb, which is known to lead to over-smoothing in the case of multi-modal distributions [20]. Furthermore, on the pedestrian trajectories distribution, we observed both high D_{JS} and $\widehat{W}$ values, indicating that VC is unable to estimate the underlying density; this is also indicated by the poor log-likelihood estimates.

Overall, while the baselines were able to achieve better performance in selected cases (e.g., MPW better than ROME in terms of $\widehat{W}$ and $\widehat{L}$ on the Two Moons distribution), they have their apparent weaknesses. Specifically, MPW achieves poor results for most metrics in the case of varying densities within the modes (Varied), while Vine Copulas obtain the worst performance across all three metrics

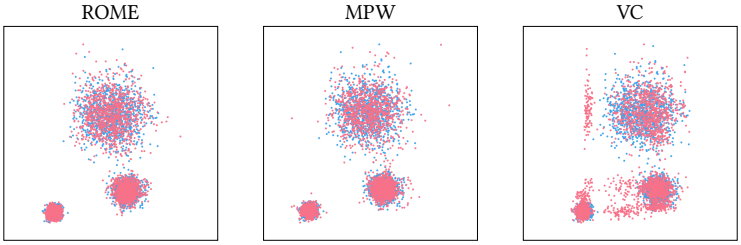


Figure 2.4: Samples obtained with ROME, MPW and VC (pink) contrasted with samples from p (blue); Varied.

Table 2.1: Baseline Comparison – marked in red are cases with notably poor performance; best values are underlined.

Distribution	$D_{\mathrm{JS}} \downarrow_0^1$ ROME	MPW	VC	$\widehat{W} \to 0$ ROME	MPW	VC	$\widehat{L} \uparrow$ ROME	MPW	VC
Aniso	0.010±0.001	0.026±0.002	0.005±0.001	−0.13±0.31	−0.60±0.13	1.91±0.91	−2.53±0.02	−2.57±0.02	−3.19±0.02
Varied	0.011±0.001	0.025±0.002	0.008±0.001	−0.13±0.20	−0.49±0.11	1.27±0.53	−4.10±0.03	−4.12±0.03	−4.29±0.03
Two Moons	0.002±0.001	0.023±0.002	0.008±0.002	1.40±0.52	−0.52±0.10	1.36±0.51	−1.02±0.01	−0.36±0.01	−0.95±0.01
Trajectories	0.008±0.002	0.016±0.001	0.743±0.005	1.03±0.22	1.24±0.24	9.30±1.30	29.32±0.02	26.09±0.02	−215.23±17.6

Table 2.2: Ablations ($\widehat{W} \to 0$, Varied): Clustering is essential to prevent over-smoothing. ROME highlighted in gray. Note that the differences between Silhouette and DBCV are not statistically significant.

Cluster.	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.	f_{GMM}
Silhouette	−0.13±0.20	−0.13±0.20	−0.13±0.19	−0.14±0.20
DBCV	−0.09±0.23	−0.09±0.23	−0.07±0.23	−0.03±0.24
No clus.	2.28±0.72	2.26±0.72	−0.17±0.26	10.53±2.54

Table 2.3: Ablations ($\widehat{L} \uparrow$, Aniso): When clustering, decorrelation and normalization improve results for distributions with high intra-mode correlation. ROME highlighted in gray.

Cluster.	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.
Silhouette	−2.53±0.02	−2.70±0.01	−2.79±0.01
DBCV	−2.56±0.02	−2.69±0.01	−2.83±0.02

Table 2.4: Ablations ($\widehat{W} \to 0$, Two Moons): Excluding normalization or using f_{GMM} as the downstream estimator is not robust against non-normal distributions. ROME highlighted in gray.

Cluster.	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.	f_{GMM}
Silhouette	1.40±0.52	1.41±0.52	3.65±0.99	4.37±1.14
DBCV	1.59±0.56	1.59±0.56	4.24±1.13	4.77±1.23
No clus.	1.82±0.60	1.84±0.60	3.17±0.89	5.29±1.34

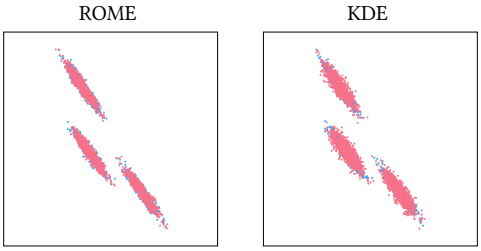


Figure 2.5: Samples generated by ROME and KDE – equivalent to ROME without clustering, decorrelation and normalization – (in pink) contrasted with samples from p (in blue); Aniso. Note that the samples by KDE are more spread out, indicating over-smoothing.

in the case of the multivariate trajectory distributions. ROME, in contrast, achieved high performance across all the test cases.

2.5.2 Ablation Studies

When it comes to the choice of the clustering method, our experiments show no clear advantage for using either the silhouette score or DBCV. But as the silhouette score is computationally more efficient than DBCV, it is the preferred method. However, using clustering is essential, as otherwise there is a risk of over-smoothing, such as in the case of multi-modal distributions with varying densities in each mode (Table 2.2).

Testing variants of ROME on the Aniso distribution (Table 2.3) demonstrated not only the need for decorrelation, through the use of rotation, but also normalization in the case of distributions with highly correlated features. There, using either of the two clustering methods in combination with normalization and decorrelation (our full proposed method) is better than the two alternatives of omitting only decorrelation or both decorrelation and normalization. In the case of clustering with the silhouette score, the full method is significantly more likely to reproduce the underlying distribution p by a factor of 1.19 (with a statistical significance of 10^{-50}, see Appendix A.2) as opposed to omitting only decorrelation, and by 1.30 compared to omitting both decorrelation and normalization. Similar trends can be seen when clustering based on DBCV, with the full method being more likely to reproduce p by a factor of 1.14 and 1.31 respectively. Results on the Aniso distribution further show that KDE on its own is not able to achieve the same results as ROME, but rather it has a tendency to over-smooth (Figure 2.5). Additionally, the ablation with and without normalization on the Two Moons distribution (Table 2.4) showed that normalization is necessary to avoid over-smoothing on

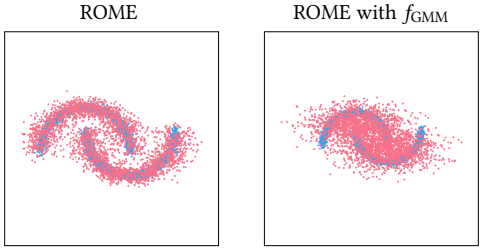


Figure 2.6: Samples generated by ROME, and ROME with f_{GMM} as the downstream estimator (in pink) contrasted with samples from p (in blue); Two Moons. Note that the samples from f_{GMM} are more spread out, which clearly displays over-smoothing.

non-normal distributions.

Lastly, investigating the effect of different downstream density estimators, we found that using ROME with f_{kNN} instead of f_{KDE} leads to over-fitting (highest D_{JS} values in Table 2.5). Meanwhile, ROME with f_{GMM} tends to over-smooth the estimated density in cases where the underlying distribution is not Gaussian (high $\widehat{W}$ in Table 2.4). The over-smoothing caused by f_{GMM} is further visualised in Figure 2.6.

In conclusion, our ablation studies confirmed that using f_{KDE} in combination with data clustering, normalization and decorrelation provides the most reliable density estimation for different types of distributions.

2.6 Conclusion

In our comparison against two established and sophisticated density estimators, we observed that ROME achieved consistently good results across all tests, while manifold Parzen windows (MPW) and vine copulas (VC) were susceptible to over-fitting and over-smoothing. For example, while MPW is superior at capturing non-normal distributions, it produces kernels with too small of a bandwidth (hence the over-fitting), which is likely caused by the selected number of nearest neighbors used for the localized kernel estimation being too small. Meanwhile, compared to VC, ROME is numerically more stable and does not hallucinate new modes in the estimated densities (Figure 2.4). Furthermore, as part of several ablation studies, we found that ROME overcomes the shortcomings of other common density estimators, such as the over-fitting exhibited by kNN or the over-smoothing by GMM. In those studies, we additionally demonstrated that our approach of using clustering, decorrelation, and normalization is indispensable for overcoming the deficiencies of KDE.

Future work can further improve on our results by investigating the integration of more sophisticated density estimation methods, such as MPW, instead of the kernel density estimator in our proposed approach to enable better performance on non-normal clusters.

Overall, by providing a simple way to accurately estimate distributions based on samples, ROME can help in better handling and evaluating probabilistic data as well as enabling more precise probabilistic inference.

Table 2.5: Ablations ($D_{\text{JS}} \downarrow_0^1$, Trajectories; values are multiplied by 10 for easier comprehension): Using f_{kNN} as the downstream estimator tends to lead to over-fitting. ROME highlighted in gray.

	Norm.				No norm.		
	Decorr.		No decorr.				
Cluster.	f_{KDE}	f_{kNN}	f_{KDE}	f_{kNN}	f_{KDE}	f_{kNN}	f_{GMM}
Silhouette	$0.084_{\pm 0.016}$	$1.045_{\pm 0.064}$	$0.777_{\pm 0.116}$	$1.808_{\pm 0.112}$	$0.015_{\pm 0.011}$	$1.887_{\pm 0.118}$	$0.032_{\pm 0.007}$
DBCV	$0.090_{\pm 0.015}$	$1.119_{\pm 0.073}$	$0.897_{\pm 0.154}$	$1.937_{\pm 0.109}$	$0.017_{\pm 0.012}$	$1.934_{\pm 0.116}$	$0.043_{\pm 0.010}$
No clus.	$0.009_{\pm 0.004}$	$0.453_{\pm 0.051}$	$0.015_{\pm 0.012}$	$1.044_{\pm 0.104}$	$0.005_{\pm 0.003}$	$1.478_{\pm 0.132}$	$0.017_{\pm 0.011}$

3

TrajFlow: Learning Distributions over Trajectories for Human Behavior Prediction

Predicting the future behavior of human road users is an important aspect for the development of risk-aware autonomous vehicles. While many models have been developed towards this end, effectively capturing and predicting the variability inherent to human behavior still remains an open challenge.

This chapter covers TrajFlow—an approach for probabilistic trajectory prediction based on a generative model known as Normalizing Flows, which is designed to fit complex distributions. We reformulate the problem of capturing distributions over trajectories into capturing distributions over abstracted trajectory features using an autoencoder, simplifying the learning task of the Normalizing Flows. TrajFlow outperforms state-of-the-art behavior prediction models in capturing full trajectory distributions in two synthetic benchmarks with known true distributions, and is competitive on the naturalistic datasets ETH/UCY, rounD, and nuScenes. Our results demonstrate the effectiveness of TrajFlow in probabilistic prediction of human behavior.

This chapter is a verbatim copy of the peer-reviewed paper [51]:

A. Mészáros, J. F. Schumann, J. Alonso-Mora, A. Zgonnikov, and J. Kober. "TrajFlow: Learning Distributions over Trajectories for Human Behavior Prediction." In 2024 IEEE Intelligent Vehicles Symposium (IV), pp. 184-191. IEEE, 2024.

Statement of contributions: Anna Mészáros contributed to the initial idea of the trajectory prediction model, developed the model itself, tested it against state-of-the-art models, and was the main author of the manuscript. Julian F. Schumann provided a testing framework which facilitated the evaluation of the models. Javier Alonso-Mora, Arkady Zgonnikov, and Jens Kober provided valuable feedback at all steps of the project.

Code pertaining the experimental setup for this chapter can be found at `https://github.com/anna-meszaros/TrajFlow-Experimet-Setup`.

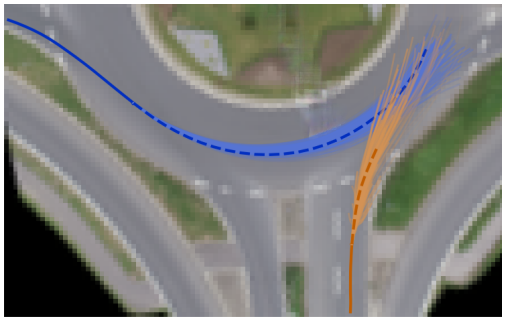


Figure 3.1: An example of the predictions generated by TrajFlow on the rounD dataset. Level of opacity indicates the likelihood of a given prediction.

3.1 Introduction

Autonomous vehicles (AVs) have become an important field of research due to many promised benefits which include, but are not limited to, improved safety, accessibility, as well as reduced traffic congestion [52–54]. Yet they are still not widespread, in big part due to their inability to effectively resolve interactions with humans [55, 56]. Being able to reliably and accurately predict human behavior would allow for more efficient and safe AV path planning [57].

However, predicting human behavior in traffic is complicated by the fact that such behavior is generally not deterministic, but instead stochastic, with potentially complex and multi-modal distributions [15]. An example of such multi-modality can be seen at roundabouts, where vehicles have the option to enter the roundabout directly or to wait for an oncoming car to pass. While these two options are the most obvious high-level behaviors, there can also be other distinct modes, such as deciding whether or not to slow down before entering the roundabout (Fig. 3.1). Such modes are scenario-dependent and may get overlooked by methods that rely on a predefined number of modes [58, 59].

Several methodologies for providing probabilistic predictions over traffic agents' future trajectories have been proposed, ranging from Gaussian Mixture Models (GMMs) [58, 59] to generative networks. Generative networks such as Generative Adversarial Networks (GANs) [60, 61], Variational Autoencoder (VAE) based networks [62–64], and diffusion models [65], are particularly interesting due to their potential to learn complex multi-modal distributions without specifying the number of expected modes, unlike methods that rely on GMMs. While these state-of-the-art approaches already achieve good results in prediction accuracy, they have the fundamental problem of being trained to reproduce the *only* true future trajectory available for each past trajectory in the dataset, thereby ignoring the underlying stochasticity of human behavior. This training approach can result in mode collapse, which is especially problematic for GAN-based methods [60, 61]. Additionally, many state-of-the-art models predict distributions at individual time steps [62, 66], ignoring the correlation between different time steps. These kinds of predictions can lead to more conservative strategies within the subsequent motion planning [22].

To overcome these issues, one promising approach is Normalizing Flows (NFs) [67, 68], which are specifically designed to learn underlying distributions in data and have been shown to have the capability of capturing multi-modal distributions. While NFs can be used to learn distributions of positions at individual time steps [69, 70], more recent work has expanded to providing distributions over complete trajectories [71–73]. Even though the above NF methods already demonstrate good qualitative results in predicting multiple future trajectories, it remains unclear how well these models

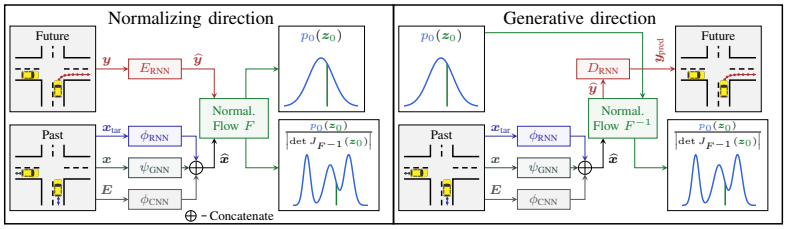


Figure 3.2: Architecture of TrajFlow. During training we use the normalizing direction in which we encode the future trajectories $\mathbf{y}$ with E_{RNN} and transform the abstracted features $\widehat{\mathbf{y}}$ to a sample $\mathbf{z}_0 = F(\widehat{\mathbf{y}})$ assumed to follow a standard normal distribution with the probability density function p_0. For inference we then use the generative direction, in which a sample $\mathbf{z}_0 \sim p_0$ is inversely transformed by the Normalizing Flow to generate the abstracted future trajectories $\widehat{\mathbf{y}} = F^{-1}(\mathbf{z}_0)$ that are decoded with D_{RNN} into the actual trajectories $\mathbf{y}_{\text{pred}}$. The likelihood of the encoded trajectory is obtained with $p_0(\mathbf{z}_0)|\det J_{F^{-1}}(\mathbf{z}_0)|^{-1}$. The encoding ϕ_{CNN} of map E, and the encoding ψ_{GNN} of social interactions are optional blocks, which can provide richer context information.

capture the true underlying distribution of the data. Furthermore, the previously mentioned NF models which predict over the complete trajectories require one to set the number of predicted time steps during their design, which might limit their applicability and usefulness in an online setting.

In response to these challenges, the main contribution of this work is *TrajFlow*—a prediction model with an improved capability for fitting distributions present in underlying training samples. The model builds on top of *FloMo* [73], which we extend with a key component in the form of a Recurrent Neural Network Autoencoder (Fig. 3.2). This extension generates an intermediate representation of the trajectories, which captures the most relevant features of the trajectories and in turn also simplifies the learning of the underlying distribution. The decoder of the Autoencoder is built in an auto-regressive manner, which additionally gives the model the flexibility to predict trajectories beyond the length of the seen training data. We validated our approach on a synthetic dataset for which we know the underlying distribution, as well as on an augmented version of the multi-modal Forking Paths dataset [34], and several popular real-world datasets (ETH/UCY [74, 75], rounD [76], and nuScenes [77]).

3.2 Background: Normalizing Flows

Normalizing Flows constitute a family of generative methods which enable exact likelihood computation. They are based on the concept of transforming distributions through a series of differentiable bijective functions into a simple known "base" distribution Z_0 – most commonly a standard normal distribution.

A number of ways for constructing flow models have been proposed [78]. One possible way is by using auto-regressive flows, consisting of a series of K normalizing layers. The main components of these layers are the conditioner c_k and the transformer τ_k, which are often accompanied by an additional permutation layer ϵ_k. The latter two functions (τ_k and ϵ_k) are bijective – and therefore invertible. In the generative direction, these functions then enable the transformation of a sample $\mathbf{z}_0$ from the base distribution Z_0 towards the desired distribution Z_K:

$$\mathbf{z}_{k+1} = \epsilon_k(\tau_k(\mathbf{z}_k;\boldsymbol{\theta}_k)), \quad \text{with} \quad \boldsymbol{\theta}_k = c_k\left(\mathbf{z}_k;\widehat{\mathbf{x}}\right),$$

where $\mathbf{z}_{k+1}$ is the result of the k-th intermediate transformation. Meanwhile, $\widehat{\mathbf{x}}$ is a conditioning input [79] that can take the form of e.g. an encoding of observations like past trajectories, static

environment, and social interactions. In the normalizing direction, F is then a composition of all K layers, where it is possible to exploit the property of c_k that $\boldsymbol{\theta}_k = c_k\left(\epsilon_k^{-1}(\mathbf{z_{k+1}}); \widehat{\mathbf{x}}\right)$:

$$F(\mathbf{z}_K) = \left(\tau_0^{-1} \circ \epsilon_0^{-1} \cdots \circ \tau_K^{-1} \circ \epsilon_K^{-1}\right)(\mathbf{z}_K) = \mathbf{z}_0$$

In the generative direction, this then allows the drawing of a sample $\mathbf{z}_K = F^{-1}(\mathbf{z}_0)$ from the desired non-normal distribution over outputs Z_K, using a sample $\mathbf{z}_0$ from Z_0. The Probability Density Function (PDF) p_K of Z_K can then also be obtained in terms of the PDF p_0:

$$\begin{aligned} p_K(\mathbf{z}_K) &= p_0(F(\mathbf{z}_K))\,|\det J_F(\mathbf{z}_K)| \\ &= p_0(\mathbf{z}_0)\,|\det J_{F^{-1}}(\mathbf{z}_0)|^{-1}, \end{aligned}$$

The absolute determinant of the Jacobian $|\det J_F(\mathbf{z}_K)|$ quantifies the relative change of volume within a small neighborhood of $\mathbf{z}_K$ when transforming it to a sample $\mathbf{z}_0$ using F. This ensures that the probability mass remains the same between the two distributions.

The parameters of F are learned by minimizing the KL-divergence between the target distribution Z_K^* with PDF $p_K^*(\mathbf{z}_K)$ and the learned distribution Z_K with the PDF $p_K(\mathbf{z}_K)$:

$$\begin{aligned} \mathcal{L} =& D_{\mathrm{KL}}[p_K^*(\mathbf{z}_K) \| p_K(\mathbf{z}_K)] \\ =& -\mathbb{E}_{\mathbf{z}_K \sim Z_K^*}\Big[\log p_0(F(\mathbf{z}_K)) + \log|\det J_F(\mathbf{z}_K)| \\ & \qquad - \log p_K^*(\mathbf{z}_K)\Big] \end{aligned}$$

With only a finite number N of samples $\mathbf{z}_{K,n}$ representing the underlying distribution Z_K^* and ignoring the constant part $\log p_K^*(\mathbf{z}_K)$, this loss can be approximated with:

$$\mathcal{L} \approx -\frac{1}{N}\sum_{n=1}^{N} \log p_0(F(\mathbf{z}_{K,n})) + \log|\det J_F(\mathbf{z}_{K,n})|.$$

3.3 Method: TrajFlow

We build up on the *FloMo* approach [73], in which NFs are employed for learning distributions directly on two-dimensional trajectories $\mathbf{y} \in \mathbb{R}^{n_O \times 2}$ defined at n_O future time steps (where $\mathbf{y} = \mathbf{z}_K$). However, attempting to learn a distribution over the trajectories directly makes it susceptible to overfitting to the variability inherent in human behavior [80] as well as noise in its measurements. Additionally, as n_O in this design is fixed, the model has a limited prediction horizon, hindering its general applicability. Furthermore, intuitively people do not observe trajectories as a series of precise positions at each time step. Instead, they perceive a trajectory more abstractly in terms of general direction, length, and shape. Therefore, learning the distribution over such abstracted characteristics might be better suited to mimic human decision making, an approach that has shown itself to be promising in improving prediction models [81, 82].

To overcome these challenges and to facilitate the learning of underlying distributions, we constructed our proposed model *TrajFlow* to let the NFs reason over trajectory abstractions rather than the raw trajectories.

3.3.1 Normalizing Flow

In our specific case, we chose to use a *Coupling Layer* for c_k, a *Rational Quadratic Spline* for τ_k, and a permutation layer ϵ_k, similar to the *FloMo* model [73]. However, unlike [73], we did not augment the trajectories in the training data. Furthermore, we also did not inject added noise into the Normalizing Flow as was done in FloMo using the β and γ hyperparameters. Lastly, we incorporated a learning rate decay lr for the training of the Normalizing Flow, as this has proven beneficial for achieving a better distribution fit.

3.3.2 Encoding Trajectories

To capture the abstracted characteristics of a trajectory, we utilized a Recurrent Neural Network Autoencoder (RNN-AE) with encoder E_{RNN} and decoder D_{RNN}. This allows us to create an abstraction of a trajectory $\widehat{\mathbf{y}} = E_{\text{RNN}}(\mathbf{y}) \in \mathbb{R}^m$. This addition results in the novel *TrajFlow* model (Fig. 3.2) that consequently learns the distribution of the encoded future trajectories $\widehat{Y}$ (with $\widehat{\mathbf{y}} = \mathbf{z}_K$) rather than the raw future trajectories Y.

Gated Recurrent Unit

The RNN-AE uses as its main component a so-called Gated Recurrent Unit (GRU) [83], one of the main RNNs used for encoding time series events. In its most basic single-layered form with embedding dimensionality M and hidden dimensionality d, it can be depicted as a function $\phi_{\text{GRU}} : \mathbb{R}^M \times \mathbb{R}^d \to \mathbb{R}^d$, which takes at time step t an input $\mathbf{a}_t \in \mathbb{R}^M$ and uses it to change its internal hidden state $\mathbf{h} \in \mathbb{R}^d$:

$$\mathbf{h}_t = \phi_{\text{GRU}}(\mathbf{a}_t, \mathbf{h}_{t-1})$$

If no hidden layer is provided at the beginning of a sequence, those can be assumed to be zero. However, *TrajFlow* employs a multi-layered version using multiple recurrent units $\phi^{(l)}_{\text{GRU}}$, with $l \in \{1, \dots, L\}$:

$$\mathbf{h}^{(l)}_t = \phi^{(l)}_{\text{GRU}}\left(\mathbf{h}^{(l-1)}_t, \mathbf{h}^{(l)}_{t-1}\right) \quad \text{with} \quad \mathbf{h}^{(0)}_t = \mathbf{a}_t$$

This can then be combined in a multilayer function $\phi_{\text{L-GRU}} : \mathbb{R}^M \times \mathbb{R}^{L \times d} \to \mathbb{R}^{L \times d}$ with $\mathbf{H}_t = \{\mathbf{h}^{(1)}_t, \dots, \mathbf{h}^{(L)}_t\}$:

$$\mathbf{H}_t = \phi_{\text{L-GRU}}(\mathbf{a}_t, \mathbf{H}_{t-1}) \tag{3.1}$$

The RNN-Encoder

In the first step of the encoder E_{RNN}, we created a transformed trajectory $\widetilde{\mathbf{y}} = \langle \widetilde{\mathbf{y}}_1, \dots, \widetilde{\mathbf{y}}_{n_O} \rangle$ with

$$\widetilde{\mathbf{y}}_t = \mathbf{y}_t - \mathbf{y}_{t-1} \tag{3.2}$$

This is based on previous results showing that displacement information is more useful for trajectory prediction tasks [84]. We then used a linear layer $\phi_{\text{em}} : \mathbb{R}^2 \to \mathbb{R}^M$ that embeds a displacement $\widetilde{\mathbf{y}}_t$. The embedded time steps are then run in sequence through a multi-layered GRU $\phi_{\text{E-L-GRU}}$ (3.1), setting $\mathbf{a}_t = \phi_{\text{em}}(\widetilde{\mathbf{y}}_t)$. Using a second linear layer $\phi_{\text{E}} : \mathbb{R}^d \to \mathbb{R}^m$, we get our final encoded trajectory $\widehat{\mathbf{y}} = \phi_{\text{E}}\left(\mathbf{h}^{(L)}_{E,n_O}\right)$.

The RNN-Decoder

Our decoder D_{RNN}, uses as its first step a liner layer $\phi_{\text{D}} : \mathbb{R}^m \to \mathbb{R}^d$ to pre-process an encoded trajectory $\widehat{\mathbf{y}}$. We then again used a multilayer GRU $\phi_{\text{D-L-GRU}}$ (Equation (3.1)) to construct a new trajectory. Here, the initial hidden states are set to $\mathbf{h}^{(l)}_{D,0} = \widehat{\mathbf{y}}$. Meanwhile, our input is auto-regressive, i.e. $\mathbf{a}_1 = \phi_{\text{D}}(\widehat{\mathbf{y}})$ and $\mathbf{a}_t = \phi_{\text{D}}\left(\mathbf{h}^{(L)}_{D,t-1}\right)$ for $t > 1$. We constructed the final displacements using a linear layer $\phi_{\text{out}} : \mathbb{R}^d \to \mathbb{R}^2$:

$$\widetilde{\mathbf{y}}_{t,\text{pred}} = \phi_{\text{out}}\left(\mathbf{h}^{(L)}_{D,t}\right)$$

As a last step, we used the cumulative sum over $\widetilde{\mathbf{y}}_{\text{pred}}$ to construct the predicted trajectory $\mathbf{y}_{\text{pred}}$ (inverting (3.2)). While we used the same hidden dimension d and embedding size M for both $\phi_{\text{E-L-GRU}}$ and $\phi_{\text{D-L-GRU}}$, we did not use any weight sharing between them.

Training

The RNN-AE is trained separately before the rest of the network with a root mean square error reconstruction loss on the reconstructed trajectories:

$$\mathcal{L}_{\text{AE}} = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{y}_n - \mathbf{y}_{\text{pred},n}\|.$$

The choice to calculate the loss on the reconstructed trajectories instead of the decoded displacements was made to penalize cumulative errors that can arise from summing over the displacements. During the later training of the remaining parts of the model, the weights of the RNN-AE were frozen.

3.3.3 Encoding Context Information

For the observations $\widehat{\mathbf{x}}$, which are used for conditioning the distributions learned by the NF, we used the target agent's past trajectory $\mathbf{x}_{\text{tar}}$, the past trajectories of all agents $\mathbf{x}$, and optionally images of the static environment E. In order to encode these pieces of information, we used the neural networks ϕ_{RNN}, ψ_{GNN}, and ϕ_{CNN} respectively and concatenated their outputs:

$$\widehat{\mathbf{x}} = \phi_{\text{RNN}}(\mathbf{x}_{\text{tar}}) \oplus \psi_{\text{GNN}}(\mathbf{x}) \oplus \phi_{\text{CNN}}(\mathbf{E})$$

The exact implementation of these components can be found in the Appendix.

3.4 Experimental Setup

We performed a number of tests, two on synthetic datasets with known ground truth distributions (Sec. 3.5) and three on real-world datasets (Sec. 3.6). To facilitate those tests, we utilized an existing benchmarking framework [57].

3.4.1 Models

We used four state-of-the-art behavior prediction models as baselines:

- *Trajectron++ (T++)* [62]—selected as it continues to act as a strong baseline in trajectory prediction tasks. At the same time, it provides a good illustration of the potential drawbacks of fitting distributions per time step.
- *PECNet* [64]—a state-of-the-art model which captures multi-modality by predicting distributions over goal points and then regressing the trajectories to them.
- *Motion Indeterminacy Diffusion (MID)* [65]—a recent diffusion-based method for probabilistic trajectory prediction. As of late, diffusion based models have been showing promise in generating probabilistic predictions [85].
- *FloMo* [73] (FM)—since we build on this model, it is most directly comparable to our approach.
- *TrajFlow without the RNN-AE (TF w/o AE)*—to showcase the importance of the RNN-AE we tested against an ablation of TrajFlow.

For the RNN-AE in *TrajFlow* we used a $L = 3$ layered GRU with a hidden dimensionality $d = 20$, embedding dimensionality $M = 20$, and latent space dimensionality $m = 20$ (Sec. 3.3.2).

For the past trajectory encoding ϕ_{RNN}, we set the parameters in accordance to those described in Appendix B. Meanwhile, where applicable we employed the same ψ_{GNN} and ϕ_{CNN} structures for *TrajFlow*, *TF w/o AE* and *FloMo*.

The learning rate decay was set to $lr = 0.98$. These same parameters were used for *TF w/o AE*.

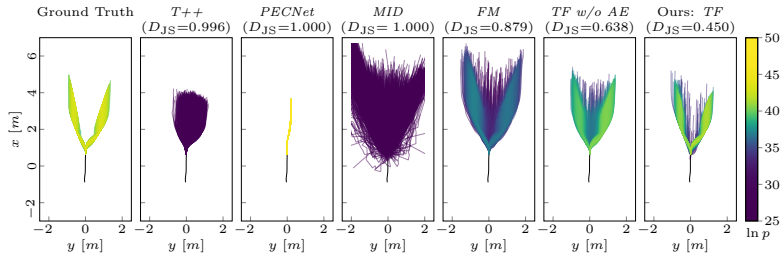


Figure 3.3: Results of the experiments on the synthetic bimodal dataset. The left-most plot depicts the ground truth distribution; the other panels are the best out of the ten distributions learned by *Trajectron++ (T++)*, *PECNet*, *Motion Indeterminacy Diffusion (MID)*, *FloMo (FM)*, an ablation of *TrajFlow* without the RNN-AE *(TF w/o AE)*, and *TrajFlow (TF)*, along with the D_{JS} values for the specific distributions that are depicted. The colors provided in the distributions are determined based on the density values obtained through density estimation on 3000 samples obtained from the respective models.

3.4.2 Metrics

To evaluate the distance of the predicted trajectories w.r.t. the ground truth trajectory we use:

- **minADE/minFDE**: Average/Final L_2 distance (measured in meters) between the best-predicted trajectory and the ground truth, based on 20 predicted samples. We chose this metric primarily to obtain an interpretable measure of how closely the predictions of the models capture the single ground truth sample available in real-world test cases, since the usefulness of distribution specific metrics such as Negative Log-Likelihood (NLL) is limited when evaluating on singular ground truth samples.

In order to obtain insight into the learned distribution over trajectories, we make use of:

- D_{JS}: Average Jensen-Shannon divergence [47] between the ground truth distribution and the learned distribution. A perfect fit of the distribution is characterized by a divergence value of 0 whereas two dissimilar distributions would result in a divergence value of 1. As this metric requires a known ground truth distribution, it is only applicable for the synthetic cases.
- **(indep.) NLL**: The average NLL of the ground truth according to the learned distribution of each individual agent. This gives us insight into the fit over the marginal distributions of the trajectories within the scene. This metric is commonly used in cases with any number of ground truth trajectories for a given scenario [62].
- **joint NLL**: The average NLL of the ground truth, based on the joint distribution of predicted trajectories for all agents in the scene. This metric gives additional information about how well the model learned the interactions between the agents in the scene.

To obtain the density estimates needed for the above metrics, we used a non-parametric density estimation approach proposed in [86] so as to ensure a more reliable comparison between models. For estimating the predicted trajectory distribution, 100 sampled trajectories were used.

3.5 Experiments: Synthetic Datasets

3.5.1 Datasets

We tested our approach on two synthetically generated datasets, one of which was generated using a single bimodal distribution, while the other is an augmented version of the Forking Paths dataset [34].

Table 3.1: Synthetic Bimodal Dataset: average results across 10 seeds

Models	minADE	minFDE	NLL	D_{JS}
T++	0.43 ± 0.03	0.66 ± 0.09	-19.08 ± 3.92	0.998 ± 0.001
PECNet	0.90 ± 0.01	$1.29 \pm 4e^{-3}$	$0.8e^{3} \pm 10.77$	$1.000 \pm 1e^{-16}$
MID	0.72 ± 0.03	0.81 ± 0.20	-3.79 ± 1.88	$1.000 \pm 5e^{-7}$
FM	0.19 ± 0.03	0.32 ± 0.05	-35.83 ± 0.72	0.916 ± 0.016
TF w/o AE	0.13 ± 0.02	0.22 ± 0.03	-38.34 ± 1.05	0.811 ± 0.081
TF (Ours)	$\underline{0.12} \pm 0.01$	$\underline{0.20} \pm 0.02$	$\underline{-39.22} \pm 0.68$	$\underline{0.683} \pm 0.132$

Table 3.2: Forking Paths: Average results across all splits & seeds.

Models	minADE	minFDE	NLL	D_{JS}
T++	0.56 ± 0.06	1.02 ± 0.13	-5.73 ± 4.21	0.985 ± 0.005
PECNet	1.05 ± 0.09	1.89 ± 0.28	$1.3e^{3} \pm 0.5e^{3}$	$1.000 \pm 1.3e^{-7}$
MID	0.89 ± 0.09	2.06 ± 0.38	-7.41 ± 2.17	$1.000 \pm 5.7e^{-5}$
FM	0.41 ± 0.04	0.70 ± 0.10	-21.86 ± 0.44	0.986 ± 0.003
TF w/o AE	$\underline{0.40} \pm 0.05$	$\underline{0.69} \pm 0.11$	-22.65 ± 0.69	$\underline{0.982} \pm 0.004$
TF (Ours)	0.42 ± 0.06	0.71 ± 0.11	$\underline{-23.69} \pm 0.99$	0.984 ± 0.004

The **synthetic bimodal dataset** was used to test the models' ability to capture the underlying distribution of the observed data. We constructed this dataset based on two recorded pedestrian trajectories with distinct directions to obtain a set of future trajectories over which we know the underlying distribution. This synthetic dataset has only a single agent and no static environment information for the observations. The trajectories were split into a past sequence $\mathbf{x}_k^*$ with $n_I = 10$ and future sequence $\mathbf{y}_k^*$ with $n_O = 14$ recorded time steps of 0.25 s each. However, for both future trajectories, we used the same past trajectory $\mathbf{x}_1^*$ so that the true output distribution is guaranteed to be bimodal. With this, we avoid the learned likelihoods becoming skewed due to slight differences in the past trajectories which could in turn make it more difficult to evaluate the predicted distributions. The set of future trajectories $\mathbf{Y}$ with 3000 samples was then created by multiplying the original two future trajectories with a random scaling factor $s \sim \mathcal{N}(1, 0.15)$:

$$\mathbf{Y} = \left\{ s_{1,i}\mathbf{y}_1^*, s_{2,i}\mathbf{y}_2^* \mid i \in \{1, \dots, 1500\} \right\}$$

The **augmented Forking Paths dataset** was used to test the methods on a more complex, multi-modal dataset. The Forking Paths dataset [34] originally included multiple human-predicted pedestrian trajectories. Within this dataset, for each past trajectory $\mathbf{x}^*$ with $n_I = 8$, there are K annotated future trajectories $\mathbf{y}_k^*$ with $n_O = 12$, recorded with a sampling frequency of 2.5 Hz. We then generated 100 augmented trajectories for each k:

$$\mathbf{y}_{k,i} = s_{k,i}\mathbf{y}_k^*\mathbf{R}_{\theta_{k,i}}^T, \text{ with } k \in \{1, \dots, K\}, i \in \{1, \dots, 100\}.$$

Here, $\mathbf{R}_\theta \in \mathbb{R}^{2\times 2}$ is a rotation matrix rotating $\mathbf{y}_k^*$ by $\theta \sim \mathcal{N}(0, \frac{\pi}{180})$, while $s \sim \mathcal{N}(1, 0.03)$ is a scaling factor.

3.5.2 Training and evaluation

Considering that the **synthetic bimodal dataset** contains a single scenario, we trained 10 instances of each model, using different random seeds, to decrease the effect of the random initialization of the models' trainable parameters.

Meanwhile, for the **augmented Forking Paths** dataset training and evaluation were performed using five-fold cross-validation, which was each repeated for 5 random seeds.

3.5.3 Results

On the synthetic bimodal dataset, we found that out of the tested models, the methods which did not employ Normalizing Flows exhibited the poorest performance. This can be attributed to different factors, from the lack of correlation between time steps (*T++*) to complete mode collapse (*PECNet*). The NF-based methods, meanwhile, were all able to capture the general shape of the underlying distribution. Out of these, our approach *TrajFlow (TF)* was able to provide the best fit. A key factor to this is the use of the RNN-AE, which becomes clear when comparing the distributions learned by *TF* with its ablation (Fig. 3.3). These qualitative results are further supported by the low NLL and D_{JS} values (Tab. 3.1) attained by *TrajFlow*.

On the augmented Forking Paths dataset, we observed that none of the models could obtain distributions identical to the ones in the evaluation set – as indicated by D_{JS} values which are close to the maximum divergence value of 1 (Tab. 3.2). This is likely due to the fact that even though each of the past inputs has a ground truth distribution over the future trajectories, similar inputs may have different output distributions which could be merged by the models and thus result in dissimilar distributions from the actual ground truth distribution. Nevertheless, NLL values show clear differences in the models' capability to capture the ground truth distributions; the Normalizing Flow methods are able to achieve better performance, with *TrajFlow* achieving the best performance.

3.6 Experiments: Real-world Datasets

3.6.1 Datasets

We tested our approach on three real-world datasets, ETH/UCY [74, 75], rounD [76], and nuScenes [77], all of which are widely used for trajectory prediction.

For testing the models on **ETH/UCY** (mostly including pedestrian crowds), we used $n_I = 8$ and $n_O = 12$ with a sampling frequency of 2.5 Hz, resulting in 3.2 s and 4.8 s of past and future data respectively. Like in the majority of prior works, we did not make use of static environment information for the sake of comparability.

For testing the models on **rounD** (drone-captured roundabouts), we set $n_I = 15$ and $n_O = 25$ with a sampling frequency of 5 Hz, which amounts to 3 s and 5 s of past and future data respectively. For our evaluation, we used the scenarios extracted from the original dataset as done in [87], which focused on the gap acceptance scenario of a vehicle entering the roundabout. There, both the trajectories of the vehicle entering the roundabout and the trajectory of the vehicle already inside the roundabout, which might be cut off by the former vehicle, have to be predicted. As it is important to predict if the other vehicle might yield when trying to plan for such scenarios, it can be necessary to predict more than $n_O = 25$ time steps. This is the case in 5.9% of the scenes in rounD, with the longest predictions requiring 35 time steps.

Lastly, on **nuScenes** (general street traffic), we used $n_I = 4$ and $n_O = 12$ with a sampling frequency of 2 Hz. For both nuScenes and rounD, full context information is available.

3.6.2 Training and Evaluation

For **ETH/UCY**, training and evaluation were performed using a leave-one-out strategy [62], using the five recording locations (ETH-univ, ETH-hotel, UCY-univ, UCY-zara01, UCY-zara02).

For **rounD**, training and evaluation were performed using five-fold cross-validation. While we evaluated the normal minADE metric based on the first 25 time steps, we also checked the models' capability to predict beyond that to test its ability of extending predictions until the point by which a yielding decision had to be reached. If a model was unable to predict beyond the 25 future time steps

Table 3.3: ETH/UCY: average results across the five locations.

Models	minADE	minFDE	indep. NLL	joint NLL
T++	0.41 ±0.17	0.66 ±0.26	−6.77 ±7.93	$0.2e^3$ ±$0.5e^3$
PecNet	2.39 ±3.00	3.34 ±3.99	$1.2e^4$ ±$2.2e^4$	$2.7e^4$ ±$4.3e^4$
MID	0.59 ±0.16	1.08 ±0.30	−0.27 ±10.21	$0.1e^3$ ±$0.2e^3$
FM	0.32 ±0.13	0.55 ±0.22	−19.65 ±4.82	−50.05 ±20.84
TF w/o AE	0.33 ±0.12	0.54 ±0.21	−18.02 ±4.11	−42.24 ±16.50
TF (Ours)	0.33 ±0.15	0.55 ±0.24	−21.05 ±5.35	−75.79 ±47.42

Table 3.4: RounD: average results across the five cross splits.

Models	minADE	minFDE	indep. NLL	joint NLL
T++	0.69 ±0.02	1.66 ±0.07	−42.04 ±1.10	−79.67 ±1.98
PECNet	1.56 ±0.11	4.49 ±0.20	$3.4e^3$ ±$0.3e^3$	$6.1e^3$ ±$0.7e^3$
MID	4.57 ±0.01	7.93 ±0.07	$0.3e^3$ ±$0.1e^3$	$0.5e^3$ ±$0.2e^3$
FM	0.85 ±0.03	1.80 ±0.06	−34.98 ±3.65	−69.35 ±4.98
TF w/o AE	0.80 ±0.01	1.72 ±0.03	−36.86 ±1.32	−71.62 ±2.43
TF (Ours)	0.67 ±0.03	1.38 ±0.09	−42.06 ±2.67	−81.23 ±4.69

that it had been trained on, the values for the remaining time steps were obtained through a simple constant velocity extrapolation.

Lastly, training and evaluation for **nuScenes** were performed using nuScenes' predefined training and validation splits. To decrease the effect of random parameter initialization, we trained 5 different versions of each model, using different random seeds.

3.6.3 Results

On ETH/UCY, we observed the same trend as in the synthetic datasets. The three NF-based methods were able to better fit the underlying data distribution compared to the methods which do not employ NFs. Out of these, *TrajFlow* clearly outperformed existing methods as well as its ablation in terms of distribution fit as captured by both NLL metrics. (Tab. 3.3) This is especially clear in the joint NLL, indicating that the learned distributions were also better able to capture the interactions among agents in a scene. We found that the state-of-the-art methods also performed worse even compared to the originally reported values, such as in the case of *T++* [62]. It is, however, important to note that this is not the first time the original *T++* results could not be replicated (see e.g. [88]), and compared to common practice, we used a stricter method for extracting testing samples, necessitating the existence of all 12 future positions.

On rounD (Tab. 3.4), *TrajFlow* was able to achieve the best results. When compared to its ablation case, there is a clear performance boost both in terms of the learned distribution but also in terms of the distance metrics. This is especially notable since in rounD the agents being predicted are vehicles, not pedestrians. As a result, the distance crossed is generally larger and thus prediction errors tend to accumulate faster. In terms of the extrapolation performance, out of all the tested models, *TrajFlow* achieved the best minADE value with 2.27 $m^{\pm 0.30}$. This was closely followed by *TF w/o AE* and *T++* with values of 2.53 $m^{\pm 0.52}$ and 2.58 $m^{\pm 0.55}$ respectively. *FloMo* achieved an error of 2.62 $m^{\pm 0.59}$, while *MID* and *PECNet* achieved an error of 10.10 $m^{\pm 1.26}$ and 19.09 $m^{\pm 1.10}$ respectively. It is worth noting that out of these methods, only *TrajFlow* and *T++* have auto-regressive capability while the remaining models require a separate extrapolation method.

Finally, on nuScenes (Tab. 3.5), *TrajFlow* outperformed all of the models in terms of the distance

Table 3.5: NuScenes: validation split results across the five seeds.

Models	minADE	minFDE	indep. NLL	joint NLL
T++	1.96 ±0.02	4.05 ±0.07	<u>10.28</u> ±0.29	<u>43.18</u> ±1.01
PECNet	26.73 ±2.30	40.04 ±2.36	$6.7e^5$ ±$1.0e^5$	$1.8e^6$ ±$2.9e^5$
MID	6.47 ±0.29	13.48 ±0.82	71.04 ±7.97	$0.3e^3$ ±23.27
FM	12.61 ±1.07	23.55 ±2.02	$0.3e^3$ ±$0.2e^3$	$0.9e^3$ ±$0.7e^3$
TF w/o AE	13.15 ±0.18	24.71 ±0.29	$0.9e^3$ ±$0.6e^3$	$1.9e^3$ ±$1.3e^3$
TF (Ours)	<u>1.86</u> ±0.06	<u>3.72</u> ±0.19	16.54 ±1.04	45.55 ±2.72

metrics minADE and minFDE. It was, however, outperformed on the distribution metrics by *T++*. What is notable, however, is that although *TrajFlow* had poorer performance than *T++* on the independent NLL, the performance of the two models on the joint NLL was comparable. This indicates that while *TrajFlow* does not capture the individual distributions as precisely as *T++* on nuScenes, the distributions learned manage to reflect the overall behavior in a scene to a similar extent as *T++*.

3.7 Conclusion

In this work, we proposed *TrajFlow*, a novel model for predicting the trajectories of human agents in traffic by applying Normalizing Flows to the latent abstraction of the future trajectories to be predicted.

Tests carried out on synthetic examples, which contained sets of grounds truths for a single input, showed that Normalizing Flow-based methods outperformed several state-of-the-art methods in terms of the distribution fits. Furthermore, among these NF-based models, *TrajFlow* had the best performance thanks to incorporating a Recurrent Neural Network-based Autoencoder.

Through evaluations on the ETH/UCY, rounD, and nuScenes datasets, we found that our model can successfully learn distributions within real-world datasets. *TrajFlow* achieved competitive results compared to state-of-the-art methods on all three datasets, and was able to clearly outperform existing methods in terms of the distribution fit on the pedestrian dataset ETH/UCY. This is particularly noteworthy since pedestrian behavior is less structured than that of vehicles and one can in turn expect a higher amount of variability.

We further observed that the introduction of an RNN-AE does not lead to a severe degradation of our predicted trajectories despite the loss of information inherent to data compression, and in fact, for rounD and nuScenes, learning over abstracted trajectory features proved to be beneficial. Tests on rounD also showed that the auto-regressive nature of our decoder provides further flexibility in terms of the possible length of the predictions and is even able to outperform state-of-the-art models in terms of prediction accuracy. This is particularly useful for cases that need a longer planning horizon to ensure safety and comfort such as when approaching a roundabout or when interacting with pedestrians close to a crosswalk.

Future work will explore ways to improve prediction quality for more structured environments such as in the case of vehicle trajectory prediction. Another point of focus will be to expand the model towards being able to provide joint predictions of all agents in a given scene. Finally, *TrajFlow*'s capacity to capture multi-modal distributions can be utilized in contingency planners which account for multiple possible outcomes [89]. The extent to which better distribution fitting is beneficial to such planners was outside of the scope of this work and should be investigated in future work.

Overall, our results indicate that *TrajFlow* compares favorably to state-of-the-art behavior prediction models in learning trajectory distributions both from highly variable data (e.g., pedestrian trajectories) as well as more constrained scenarios (such as in the case of vehicle trajectories). This

model thus has potential for application in settings where AVs have to navigate in settings with human road users in order to generate more natural and safer plans.

4

Studying the Effect of Explicit Interaction Representations on Learning Scene-level Distributions of Human Trajectories

Effectively capturing the joint distribution of all agents in a scene is relevant for predicting the true evolution of the scene and in turn providing more accurate information to the decision processes of autonomous vehicles. While new models have been developed for this purpose in recent years, it remains unclear how to best represent the joint distributions particularly from the perspective of the interactions between agents. Thus far there is no clear consensus on how best to represent interactions between agents; whether they should be learned implicitly from data by neural networks, or explicitly modeled using the spatial and temporal relations that are more grounded in human decision-making.

This chapter studies various means of describing interactions within the same network structure and their effect on the final learned joint distributions. Our findings show that more often than not, simply allowing a network to establish interactive connections between agents based on data has a detrimental effect on performance. Instead, having well defined interactions (such as which agent of an agent pair passes first at an intersection) can often bring about a clear boost in performance.

This chapter is a verbatim copy of the peer-reviewed paper [90]:

***A. Mészáros**, J. Alonso-Mora, and J. Kober. "Studying the Effect of Explicit Interaction Representations on Learning Scene-level Distributions of Human Trajectories." To appear in 2026 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2026.*

Statement of contributions: Anna Mészáros conceptualized, developed, and tested the models against state-of-the-art models, and was the main author of the manuscript. Javier Alonso-Mora, and Jens Kober provided valuable feedback at all steps of the project.

4.1 Introduction

Autonomous vehicles (AVs) promise to bring about safer and more accessible roads [52, 53]. However, for an AV to navigate in an environment with people, from drivers to vulnerable road users such as cyclists and pedestrians, AVs require the capacity to anticipate what the people around them will do. Predicting human behavior in traffic, however, is complicated by the fact that such behavior is generally not deterministic, but stochastic, taking the form of potentially complex and multi-modal distributions [15].

A possible way to approach predicting the future trajectories of agents is by predicting each individual agent in the scene based on context information such as the past trajectories of agents in the scene and static environment information [62]. These types of prediction models provide what are formally known as marginal predictions. However, a drawback of these types of models is that they fail to account for interactions between agents in the future, resulting in misinterpretations of the actual likelihood of certain outcomes – e.g. two vehicles colliding – when observing the evolution of a scene as a whole [91]. Recently, new prediction models have been developed to address the matter of joint predictions of future trajectories of all agents in a scene. A number of these models leverage multi-head attention or transformer networks to capture the interaction between agents and jointly predict their future trajectories [92, 93]. Meanwhile, other models utilize learned interaction graphs to capture interactions between agents and factorize the joint distribution, in turn simplifying the learning process such as in the FJMP model [94].

The way in which interactions between agents should be provided in machine learning approaches remains under-explored, despite being a key factor in human trajectory prediction [95]. Often, these interactions are captured implicitly by the network itself [92, 93]. In other cases, agents are connected by fully connected graphs whose edges carry agents' relative distance, velocity, or direction [96–98]. Other works capture interactions by constructing agent cliques [99, 100]. However, these approaches only jointly predict agents within cliques and ignore inter-clique interactions. Existing extensions remain limited to homogeneous groups (typically pedestrians) [101]. Meanwhile other approaches leverage spatial heuristics such as the crossing of future trajectories [94, 102], hypothetical conflicts [103, 104], collision risk [105], movement direction (i.e whether agents are moving towards or away from each other) [106], Euclidean distance [63, 107], or social forces [108] to provide more structure for learning these interactions. More abstract heuristics such as feature similarity [109] or correlation [110] have also been used to guide interaction learning. While prior work has explored agent interaction modeling [111], these approaches are largely limited to marginal prediction models with homogeneous agents. To our knowledge, no work has yet investigated which interaction representation is the most beneficial for machine learning models for joint human trajectory prediction, particularly for heterogeneous agents.

In this study, we examine how the chosen interaction representation affects not only the accuracy of individual trajectory predictions, but also the model's ability to capture distributions over multiple plausible future trajectories, which is crucial for risk-aware motion planning [112]. A promising group of models for capturing complex and even multi-modal distributions from data are Normalizing Flows (NFs) [68]. This family of methods has already been successfully applied for predicting marginal distributions over agent trajectories [51, 73]. Nevertheless, expanding these approaches for predicting joint distributions of all agents in a scene is not straightforward due to the bijective transformation functions that make up the core of NFs which require the dimensionality of the input to a NF to be kept constant. This results in complications since the number of agents in a scene often changes. A naive solution to this problem would be to introduce dummy agents and only focus on a fixed number of agents in a scene. However, this can lead to a number of issues, such as wasted computational resources if too many dummy agents are used to ensure the maximum number of agents is covered. Alternatively, one could use heuristics to determine the most important agents to predict, which may however result in relevant agents being overlooked, particularly in more crowded scenarios. In

response to these challenges we leverage the idea of factorizing the joint distribution in accordance to a directed acyclic graph (DAG) which captures the direction of interaction between agents.

The contributions of our work are two-fold. Firstly, we propose a Graph-based Motion Prediction (GMoP) model structure based on normalizing flows to capture joint distributions over the trajectories of all agents in a scene which remains flexible to the varying number of agents from scene to scene. Secondly, we use this model structure to study the manner in which interactions between heterogeneous agents [1] might be modeled on the level of the DAG and how these affect the final learned distributions. We perform the study on four popular real-world driving datasets (Argoverse [113], INTERACTION [114], nuScenes [77], and rounD [76]).

4.2 Background: Normalizing Flows

Normalizing Flows are a generative method, capable of capturing complex distributions. NFs are based on the concept of transforming distributions through a series of differentiable bijective functions into a simple known "base" distribution Z_0 – most commonly a standard normal distribution. One type of flow model is the conditional auto-regressive NF [79], consisting of a series of K normalizing layers, and conditioned by a feature vector C that can take the form of, e.g., an encoding of observations like past trajectories, static environment, and social interactions.

In the normalizing direction, the normalizing layers make up the normalizing flow function F which transforms samples z_K from the desired distribution Z_K into the base distribution Z_0. In the generative direction, it is then possible to draw a sample $\mathbf{z}_K = F^{-1}(\mathbf{z}_0)$ from the desired non-normal distribution over outputs Z_K, using a sample $\mathbf{z}_0$ from Z_0. The Probability Density Function (PDF) p_K of Z_K can then also be obtained in terms of the PDF p_0:

$$\begin{aligned} p_K(\mathbf{z}_K) &= p_0(F(\mathbf{z}_K))\,|\det J_F(\mathbf{z}_K)| \\ &= p_0(\mathbf{z}_0)\,|\det J_{F^{-1}}(\mathbf{z}_0)|^{-1}, \end{aligned}$$

The absolute determinant of the Jacobian $|\det J_F(\mathbf{z}_K)|$ quantifies the relative change of volume within a small neighborhood of $\mathbf{z}_K$ when transforming it to a sample $\mathbf{z}_0$ using F. This ensures that the probability mass remains the same between the two distributions. The parameters of F are learned by minimizing the negative log-likelihood over a finite number N of training samples $z_{K,n}$:

$$\mathcal{L} \approx -N^{-1}\sum_{n=1}^{N} \log p_0(F(\mathbf{z}_{K,n})) + \log|\det J_F(\mathbf{z}_{K,n})|.$$

[1] Scenes can include a combination of up to four different agent types – vehicles, motorcyclists, cyclists and pedestrians.

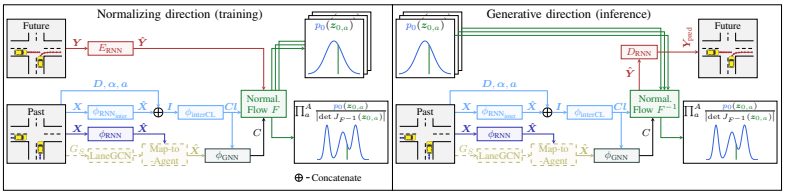


Figure 4.1: Architecture of GMoP. During training we use the normalizing direction in which we encode the future trajectories $\mathbf{Y}$ with E_{RNN} and transform the abstracted features $\hat{\mathbf{y}}_a$ to a sample $\mathbf{z}_{0,a} = F(\hat{\mathbf{y}}_a)$ assumed to follow a standard normal distribution with the probability density function p_0. For inference we then use the generative direction, in which a sample $\mathbf{z}_{0,a} \sim p_0$ is inversely transformed by the Normalizing Flow to generate the abstracted future trajectories $\hat{\mathbf{y}}_a = F^{-1}(\mathbf{z}_{0,a})$ that are decoded with D_{RNN} into the actual trajectories $\mathbf{Y}_{\text{pred}}$. The likelihood of the encoded scene-level predictions is obtained with $\prod_a^A p_0(\mathbf{z}_{0,a}) |\det J_{F^{-1}}(\mathbf{z}_{0,a})|^{-1}$. The encoding ϕ_{CNN} of map E, and the scene graph processing modules LaneGCN and Map-to-Agent are optional modules, which can provide richer context information.

4.3 GMoP – Graph-based Motion Prediction

The goal of our approach is to predict a distribution over future agent trajectories $\mathbf{Y} \in \mathbb{R}^{n_A \times n_O \times 2}$, given past observations of all agents $\mathbf{X} \in \mathbb{R}^{n_A \times n_I \times 2}$ and static environment information in the form of scene graphs, which together constitute the context C. Here, n_A denotes the number of agents, and n_I and n_O denote the number of input and output timesteps, respectively.

An important thing to consider when working with normalizing flows is that the number of features need to be preserved between the distribution that one is trying to fit and the base (i.e. latent) distribution of the NF. To circumvent this issue we leverage the idea of factorizing the joint distribution over all agents in a scene based on a learned interaction graph in the form of a directed acyclic graph (DAG) as introduced in FJMP [94]. Factorizing the problem in accordance to a DAG enables us to break the problem of learning a complete joint distribution into learning a series of smaller conditional distributions in the form

$$p_K(\mathbf{Y}|C) = \prod_a^A p_K(\mathbf{y}_a|\mathbf{Y}_{P,a}, C),$$

where $\mathbf{Y}_{P,a}$ are the future trajectories of the parent nodes P of a given agent, a. As a result, the feature dimension of $\mathbf{z}_0$ will always be equal to that of $\mathbf{y}_a$ (where $\mathbf{y}_a = \mathbf{z}_K$) for each conditional distribution $p_K(\mathbf{y}_a|\mathbf{Y}_{P,a}, C)$, irrespective of the number of agents in the scene.

To obtain this factorization we first need a DAG, based on an interaction graph for the agents in a given scene. We take this opportunity to investigate the manner in which the interaction graph can be constructed.

4.3.1 Learning the Interaction Graph

To predict the interaction graph we draw inspiration from FJMP [94] and employ a two layer MLP classifier ϕ_{interCl} with an embedding layer $\phi_{\text{em}} : \mathbb{R}^f \to \mathbb{R}^M$ that has an input feature dimensionality of f and embedding dimensionality M, and a classification layer $\phi_{\text{cl}} : \mathbb{R}^M \to \mathbb{R}^3$ followed by a Softmax activation to ensure the probabilities of the three classes (m influences n, n influences m, and no interaction) sums to one.

The classifier takes as input the encoded past trajectories of an m-n agent-pair $\hat{\mathbf{x}}_m$, and $\hat{\mathbf{x}}_n$ - obtained by passing the past trajectories $\mathbf{x}_m$ and $\mathbf{x}_n$ through an RNN $\phi_{\text{RNN}_{\text{inter}}}$ - a one-hot encoding of the agents' types, a distance vector $d_{m,n} = x_{H_m} - x_{H_n}$, and lastly the angle $\alpha_{m,n}$ between the agents' distance vector $d_{m,n}$ and the source agent's (agent m in this case) final displacement $\tilde{x}_m = x_{H_m} - x_{H-1_m}$

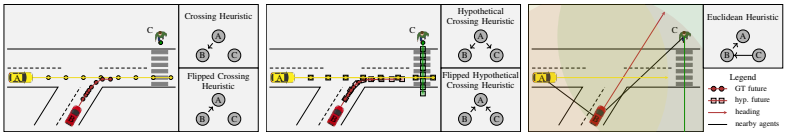


Figure 4.2: Illustration of the interaction graphs obtained using the (flipped) crossing, (flipped) hypothetical crossing and Euclidean heuristics. The resulting interaction graphs effectively represent factor graphs from which the joint distribution can be obtained. For example, in the case of the hypothetical crossing heuristic the joint distribution is factorized as $P(A)P(B|A)P(C|A)$ while in its flipped version, it is factorized as $P(B)P(C)P(A|B,C)$. Note, for the hypothetical crossing scenario, the hypothetical future is only used to check whether agents' paths would have crossed had there been no interaction between them. The direction of influence, i.e. interaction, is determined by which agent's GT future reached the hypothetical crossing point first.

obtained as

$$\alpha = \operatorname{acos}\left(\frac{d_{m,n}\cdot\tilde{x}_m}{\|d_{m,n}\|\cdot\|\tilde{x}_m\|}\right).$$

The angle α was defined in such a manner to capture interactions between agent pairs in terms of whether agent m is approaching or moving away from agent n. The agent pair information, observed from the perspective of agent m, is concatenated into a single vector $i_{m,n} \in \mathbb{R}^f$. This is then passed through the classifier $cl_{m,n} = \phi_{\text{interCl}}(i_{m,n}) = \phi_{\text{cl}}(\phi_{\text{em}}(i_{m,n}))$ to obtain the classification $cl_{m,n}$ of the interaction between the m-n agent-pair. An overview of the complete architecture used can be found in Fig. 4.1.

4.3.2 Interaction Graph Heuristics

Euclidean distance is one of the most commonly used heuristics for establishing interactions between agents. Agents are considered to be interacting if they are within a distance ϵ of each other in the last past timestep. Since we need uni-directional edges between agents for factorizing the joint distribution, we expand this heuristic to consider the angle $\phi_{m,n}$ at which an agent m sees an agent n. This angle is obtained as:

$$\phi_{m,n} = |\operatorname{atan2}(s_{n,y} - s_{m,y}, s_{n,x} - s_{m,x}) - \gamma_m| \in [0,\pi]$$

where $s_{m,x}$, $s_{n,x}$, $s_{m,y}$, $s_{n,y}$ represent the x and y positions of agents m and n at the last past timestep H, and γ_m is the heading angle of agent m. We use an angle of $\phi_{m,n} \in [0,\pi]$ since we do not distinguish if an agent is seen on the left or right from the view of the observing agent m. An agent n is influenced by an agent m, if the angle $\phi_{n,m}$ at which agent n sees agent m is smaller than $\phi_{m,n}$. This then translates to a directed edge going from agent m to n. We do this under the assumption that an agent focuses more on another agent the closer the other agent is to the center of their field of view, i.e. aligned with their heading direction. We further weigh the interaction using edge weights defined as:

$$w_{m,n} = \frac{\epsilon - d(x_{H_m}, x_{H_n})}{\epsilon},$$

where $d(x_{H_m}, x_{H_n})$ is the Euclidean distance between the positions x_{H_m} and x_{H_n} of agents m and n at the last past timestep H. This weighing captures the aspect that the further away an agent m is from agent n, the less of an effect this agent has on agent n. It should be noted that for this heuristic, no training is needed, since the construction of the graph follows directly from the above formalization.

The crossing heuristic used in FJMP [94], determines the type of interaction between agent pairs from the agents' ground truth future trajectories. More concretely, they perform a crossing

check between pairs of trajectories. If agent m reaches the crossing point before agent n, then agent m influences agent n. If there is no crossing point between the trajectories then there is no influence and in turn no edge between the two agents in the interaction graph. This information was then used to pre-train the interaction graph.

In our approach we formalize this heuristic in the following manner. Given the trajectory pairs $(\mathbf{y}_m, \mathbf{y}_n)$ we calculate a distance matrix D of the form:

$$D = \begin{bmatrix} d(y_{0_m}, y_{0_n}) & \dots & d(y_{0_m}, y_{T_n}) \\ \vdots & \ddots & \vdots \\ d(y_{T_m}, y_{0_n}) & \dots & d(y_{T_m}, y_{T_n}) \end{bmatrix}$$

for all timesteps of the future trajectories, where $d(y_{t_m}, y_{t_n})$ is the Euclidean distance between the trajectories $\mathbf{y}_m$ and $\mathbf{y}_n$ at timestep $t \in [0, T]$. To establish whether the trajectories cross at any point in time, we check whether any of the distances in D fall under a specified threshold ϵ_m^a, dependent on the type of agent a and their average width, since the distances are calculated based on agent center-points. More specifically, a crossing is detected when $d(y_{t_m}, y_{t_n}) \leq \epsilon_m^a$. We then check the timesteps for the first detected crossing. If $t_{c,m} < t_{c,n}$, then agent m influences agent n which in the interaction graph is a directed edge going from agent m to agent n. Conversely if $t_{c,m} > t_{c,n}$, the directed edge points from agent n to agent m. If no crossings are detected, there is no interaction between the agents and as such no edge.

A shortcoming of this heuristic is that it assumes agent paths need to cross for an interaction to occur. However, interactions can also occur in the form of preventing an agent from carrying out their intended behavior. A simple example of this is a pedestrian standing on the side of a road, intending to cross yet being forced to wait due to vehicles passing by.

Hypothetical crossing heuristic is a variant of the above heuristic which we define to address the aforementioned shortcoming. For this we extrapolate a given ground truth future trajectory based on the agent's velocity $\dot{x}_H$ and heading θ_H at the last timestep of the past trajectory $\mathbf{x}$. To ensure realistic trajectories, we use the ground truth future trajectory $\mathbf{y}$ as a guideline, so as to maintain the overall shape of the trajectory while speeding it up. The manner in which the speed up is performed is based on two factors. The first is a check for $\dot{x}_H^a < v^a$, where v^a is the average speed for a given agent type a. The average speeds for the agent types are based on the statistics obtained from the nuScenes [77] dataset as a guideline for normal speeds in urban environments. This check captures both agents which are at a standstill as well as agents forced to move at a slower pace than normal. For these agents, we extrapolate the trajectories such that

$$\forall t \ \dot{y}_t^a < v^a \Rightarrow \dot{y}_t^a = v^a.$$

This ensures, that if at any point in the future the agent's trajectory moves at less than average velocity, it will be sped up to reach the average velocity for its agent type. In the case that $\dot{x}_H^a > v^a$, the extrapolation is performed such that

$$\forall t \ \dot{y}_t^a < \dot{x}_H^a \Rightarrow \dot{y}_t^a = \dot{x}_H^a.$$

This captures the cases where an agent was forced to slow down in the future, potentially due to surrounding agents. Based on these extrapolated trajectories, a crossing check such as the one performed for the crossing heuristic is applied. However, the result of the crossing check is not directly used to define the interactions. Instead it is only used as an indication of whether the agents' trajectories might have crossed had the two agents ignored each other. If so, we then check which agent reached the crossing point first in reality, and use this to define the direction of interaction analogously to the definition of the crossing heuristic.

Flipped crossing and flipped hypothetical crossing heuristics are variants of the crossing and hypothetical crossing heuristics in which we flip the direction of the interaction edges. From a mathematical perspective, both would be valid factorizations of a joint distribution since $P(A,B) = P(A|B)P(B) = P(B|A)P(A)$. What we aim to study is whether the semantic meaning of the interaction edges can affect the training of the network.

In the case of either the **(flipped) crossing** or **(flipped) hypothetical crossing heuristic** the interaction classifier (Sec. 4.3.1) is pretrained using the interaction classes for every agent-pair according to the used heuristic. A weighted cross-entropy loss is used to account for class imbalance between interacting and non-interacting agent-pairs

$$\mathcal{L} = -\sum_{c}^{C} w_c y_c \log(p_c)$$

where w_c is the class weight, y_c the true class probability which is set to 1 for the correct class, and p_c is the predicted class probability. After training, the interaction classifier and the RNN-encoder $\phi_{\text{RNN}_{\text{inter}}}$ for the past trajectories are frozen.

No heuristic is a further case we investigate since there is always the potential that a chosen heuristic is not the correct one for a given task. In this case, the neural network has to learn the best graph connections purely from data in accordance to the loss for fitting the final distribution as provided in Sec. 4.3.3.

Additional information in the form of edge weights for the four crossing heuristics and for the case of no heuristics is provided based on the predicted classification probabilities for the three possible cases (no interaction, agent m affects agent n, and agent n affects agent m). The structure of the interaction classifier is described in Sec. 4.3.1.

Independence between agents is an assumption used as a baseline within our method. Although we assume there are no interactions in this case, the training loss from Sec. 4.3.3 still optimizes for fitting a joint distribution.

4.3.3 Fitting the Distribution

It was previously shown that a better fitting over distributions can be achieved by leveraging an auto-encoder to obtain an abstracted representation of the trajectories over which a distribution should be learned [51]. We apply the same strategy of first transforming the trajectories from positions to displacements with $\tilde{y}_t = y_t - y_{t-1}$, and then encoding these trajectories using the RNN-encoder E_{RNN} of a pretrained RNN-auto-encoder (RNN-AE) to obtain abstracted trajectories $\hat{\mathbf{y}}$. The abstracted trajectories are then passed to the NF. The conditional distribution being learned is thus $p_K(\hat{\mathbf{y}}_a|\hat{\mathbf{Y}}_{P,a}, C)$. For our specific implementation, we aggregate the future predictions of the parent nodes of agent a through sum-pooling to obtain $\hat{\mathbf{Y}}_{P,a}$.

Since we are fitting a joint distribution, we employ a loss over the joint distribution of all agents in a scene:

$$\mathcal{L} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{a}^{A} \log p_0(F(\hat{\mathbf{y}}_{a,n})) + \log|\det J_F(\hat{\mathbf{y}}_{a,n})|$$

During the final inference, the predicted abstracted trajectories $\hat{\mathbf{y}}$ are decoded using the RNN-decoder D_{RNN} of the RNN-AE to obtain the final predicted trajectory $\mathbf{y}_{pred}$.

4.3.4 Encoding Context Information

For encoding the context information we use a series of neural networks. First, we use an RNN encoder ϕ_{RNN} to encode the past trajectories $\mathbf{X}$ of all agents in the scene, giving us $\hat{\mathbf{X}}$. If a scene-graph

G_S is provided by the dataset, this graph is processed by LaneGCN [115] and fused with the encoded past trajectories $\hat{\mathbf{X}}$ using the Map-to-Agent network from [115]. The enriched encoding is then further passed through a Graph Neural Network (GNN) ϕ_{GNN} to encode the interactions between the agents in the scene. For this GNN we utilize the same graph structure as the one provided by the previously described interaction heuristics. The output of the GNN is the resulting context vector $C = \phi_{\text{GNN}}(\hat{\mathbf{X}})$.

4.4 Experimental Setup

4.4.1 Models

To study the effect of the interaction graph's structure on the final prediction performance we compare the seven versions of our method [2]:

- GMoP assuming independence between agents (indep)
- GMoP without heuristics (w/o H)
- GMoP with Euclidean distance heuristic (euclH)
- GMoP with crossing heuristic (crH)
- GMoP with hypothetical crossing heuristic (hcrH)
- GMoP with flipped crossing heuristic (flip crH)
- GMoP with flipped hypothetical crossing heuristic (flip hcrH).

We use three state-of-the-art scene-level trajectory prediction models as baselines:

- ADAPT [92] - efficient joint trajectory prediction model.
- AutoBots [93] - transformer based joint prediction model.
- FJMP [94] - an architecture which learns a directed acyclic interaction graph to factorize the future joint distribution for a given scene.

It is important to note that ADAPT and FJMP only provide a fixed number of deterministic joint predictions, where each prediction is a mode. Meanwhile, AutoBots outputs a Mixture Model from which predictions can be sampled.

4.4.2 Datasets

We test the methods on four widely used naturalistic datasets with heterogeneous agents, Argoverse [113], INTERACTION [114], nuScenes [77], and rounD [76].

For testing the models on **Argoverse**, we use $n_I = 50$ input timesteps and $n_O = 60$ output timesteps with a sampling frequency of 10 Hz, resulting in 5 s and 6 s of past and future data respectively. For **INTERACTION**, we use $n_I = 10$ and $n_O = 30$ with a sampling frequency of 10 Hz, resulting in 1 s of past and 3 s of future data. For **nuScenes**, we use $n_I = 4$ and $n_O = 12$ with a sampling frequency of 2 Hz, resulting in 2 s of past and 6 s of future data.

For the three datasets above, the datasets were split based on their respectively provided training and validation sets.

Lastly, for the **rounD** dataset, we use $n_I = 15$ and $n_O = 25$ with a sampling frequency of 5 Hz, resulting in 3 s and 5 s of past and future data respectively. For our experiments, we use the scenarios extracted from the original dataset as done in [87], which focused on the gap acceptance scenario of a vehicle entering the roundabout. This in turn results in a more interactive subset of the original dataset. For this dataset, we employ a leave-one-out training paradigm, in which models would be

[2] Code at: `https://github.com/anna-meszaros/GMoP-Experiment-Setup`

trained on all but one recording location. This results in three unique splits, one for each of the three locations provided in rounD.

To decrease the effect of random parameter initialization on our final analysis of the results, all of the models are trained using 5 different random seeds.

4.4.3 Metrics

We evaluate the predictions using the following metrics:

- **joint minADE/joint minFDE** - Average/Final L_2 distance (in meters) between the best-predicted trajectories for all agents in the scene and the ground truth, based on 6 predicted samples. The number of samples was chosen based on the fact that both ADAPT and FJMP provide 6 predictions. We chose this metric primarily to obtain an interpretable measure of how closely the predictions of the models capture the single ground truth sample available in real-world test cases, since the usefulness of distribution specific metrics such as Negative Log-Likelihood (NLL) is limited when evaluating on singular ground truth samples. Furthermore, since two of the state-of-the-art methods only predict a fixed number of deterministic trajectories instead of distributions this metric also allows us to better compare our methods to the state-of-the-art.
- **joint NLL** - The average NLL of the ground truth, based on the joint distribution of predicted trajectories for all agents in the scene. This metric provides us with insight on how well the learned distributions fit the ground truth data. To obtain the density estimates needed for the NLL metric, we use a non-parametric density estimation approach proposed in [86] so as to ensure a more reliable comparison between models. For estimating the predicted trajectory distribution, up to 100 sampled trajectories were used.

For the sake of brevity the above metrics will be referred to as minADE, minFDE, and NLL. For performing the experiments we utilize the STEP [116] benchmarking framework. Note that joint metrics are inherently more challenging than the corresponding single-agent metrics, as they require accurate predictions for all agents in a scene simultaneously.

4.5 Results

Comparing the performance of the different GMoP variants to the state-of-the-art models, we can see that in terms of the distance metrics minADE/minFDE our models fall in the middle of the performance range attained by the three baselines. It is important to note that our models are optimized for achieving the best possible distribution fits across the dataset rather than trying to predict a discrete set of trajectories for a given scenario. This is also reflected in the NLL metric, where the GMoP variants are able to consistently achieve better results, outperforming even the AutoBots model which also optimizes for distribution fit.

Looking more closely at versions of GMoP, we can quickly see that out of the different versions there is no one choice that reliably works well for all situations. Out of the two large-scale datasets - Argoverse and INTERACTION - we can see that on Argoverse (Tab. 4.1), all of the heuristics save for the **Euclidean heuristic** achieve comparable performance in terms of the distance metrics. We however observe a difference in performance when looking at the distribution fit, in which case the **flipped crossing heuristic** achieved the best average performance followed closely by the **crossing heuristic** and the **independence** assumption.

Meanwhile, on the INTERACTION dataset we observe that the variants of our model perform comparably to each other. The worst performance out of the variants, both in terms of average performance as well as higher standard deviation can be seen for the cases with **no heuristic** as well as the **(flipped) crossing heuristic**. At the same time, some of the best performance –

Table 4.1: Argoverse: average results across five random seeds. Per metric, the best performing model is underlined, while bold values highlight the best performing version of GMoP.

Models	minADE	minFDE	NLL
ADAPT	1.89 ±0.07	4.56 ±0.13	$4.88e^3$ ±357.14
AutoBots	4.84 ±0.52	6.22 ±0.45	$2.98e^5$ ±7.04e^4
FJMP	2.38 ±0.77	5.96 ±2.17	$8.08e^4$ ±6.50e^4
GMoP w/o H	4.48 ±4.56	10.65 ±10.24	146.69 ±324.02
GMoP indep	2.21 ±0.05	5.53 ±0.11	−142.95 ±73.12
GMoP euclH	5.74 ±7.06	13.81 ±16.64	493.68 ±979.89
GMoP crH	2.21 ±0.10	5.48 ±0.17	−149.62 ±213.09
GMoP hcrH	**2.18** ±0.02	**5.44** ±0.07	178.27 ±358.92
GMoP flip crH	2.23 ±0.02	5.58 ±0.05	**−161.22** ±136.80
GMoP flip hcrH	2.24 ±0.05	5.60 ±0.11	236.73 ±312.31

Table 4.2: INTERACTION: average results across five random seeds. Per metric, the best performing model is underlined, while bold values highlight the best performing version of GMoP.

Models	minADE	minFDE	NLL
ADAPT	1.21 ±0.10	2.52 ±0.16	$5.27e^3$ ±204.91
AutoBots	0.55 ±0.04	1.49 ±0.05	$1.83e^3$ ±311.50
FJMP	0.66 ±0.25	1.82 ±0.61	$2.22e^3$ ±1.06e^3
GMoP w/o H	0.74 ±0.12	2.23 ±0.35	−632.79 ±9.42
GMoP indep	**0.66** ±0.06	**2.00** ±0.16	−640.51 ±4.77
GMoP euclH	0.70 ±0.10	2.08 ±0.27	−624.13 ±2.72
GMoP crH	0.74 ±0.11	2.28 ±0.38	−636.38 ±6.05
GMoP hcrH	0.68 ±0.05	2.06 ±0.16	**−640.79** ±0.97
GMoP flip crH	0.78 ±0.13	2.39 ±0.41	−633.97 ±5.80
GMoP flip hcrH	0.70 ±0.06	2.09 ±0.15	−639.98 ±1.98

both in terms of average performance as well as low variability across seeds – is achieved by the **(flipped) hypothetical crossing heuristic** which accounts for agents not being able to execute their intentions due to the actions of the agents around them. This is understandable since INTERACTION is a dataset featuring highly interactive and crowded traffic scenarios, meaning that interactions extend beyond the aspect of which agent crosses paths with another agent first. At the same time, it is noteworthy that similar performance is achieved using the **independence** assumption.

Looking now at rounD Location 0, we can see that the heuristics (**Euclidean distance heuristic**, **(flipped) crossing heuristic**, and **(flipped) hypothetical crossing heuristic**) all perform reasonably well in terms of distance metrics. The **(flipped) hypothetical crossing heuristic** additionally achieves better NLL compared to the other heuristics. Meanwhile, the **independence** assumption, while obtaining worse performance in terms of distance metrics, obtained substantially better performance in terms of NLL. Similar observations had been made in [51], where better performance in distance metrics did not always imply a better distribution fit and vice versa. It is worth noting that the generally poorer performance of both our models and the state-of-the-art baselines on Location 0, compared to the other two locations can be attributed to the difference in dataset sizes generated by the splits, with Location 0 having only 971 training instances, while Locations 1 and 2 were trained with 11220 and 11805 instances respectively. Nevertheless, the better performance of

Table 4.3: NuScenes: average results across five random seeds. Per metric, the best performing model is underlined, while bold values highlight the best performing version of GMoP.

Models	minADE	minFDE	NLL
ADAPT	3.54 ±0.21	7.26 ±0.15	508.04 ±115.78
AutoBots	<u>3.43</u> ±0.07	<u>7.06</u> ±0.11	$1.01e^3$ ±124.06
FJMP	5.68 ±4.97	11.53 ±9.61	$1.56e^5$ ±$3.01e^5$
GMoP w/o H	5.48 ±2.16	12.15 ±4.29	446.70 ±202.09
GMoP indep	4.43 ±0.11	10.19 ±0.27	542.31 ±264.23
GMoP euclH	5.79 ±2.92	13.28 ±6.70	371.91 ±120.77
GMoP crH	4.31 ±0.10	9.80 ±0.25	<u>**348.83**</u> ±81.07
GMoP hcrH	**4.28** ±0.09	**9.74** ±0.18	451.60 ±111.90
GMoP flip crH	**4.28** ±0.12	9.77 ±0.26	353.31 ±63.52
GMoP flip hcrH	4.31 ±0.10	9.85 ±0.20	510.90 ±112.76

the **(flipped) hypothetical crossing heuristic** compared to other versions of our model on this Location, speaks for the fact that adequate heuristics can guide the learning of a model better in the presence of limited training data. At the same time the better distribution fit achieved when using the **independence** assumption indicates that attempting to establish structure through heuristics can be highly detrimental to the model if the heuristics fail to capture the interactions between agents. Meanwhile, on Location 1 we observe comparable performance between variants of our model with the exception of the model with **no heuristics**. This model achieved the worst performance in terms of distance metrics both in regards to the average metric values as well as a much higher level of variability. On Location 2 we see comparable performance between all variants of our model. The main difference in performance on this location is the performance in regards to the distribution fit, with the best NLL values being obtained through the **independence** assumption.

Meanwhile on nuScenes we once more observe that leveraging heuristics for defining the interactions between agents can help improve performance. Specifically, we observe the best performance in terms of distance metrics when using the **(flipped) crossing heuristic** and **(flipped) hypothetical crossing heuristics**, and to a lesser extent when using the **independence** assumption. At the same time, the **(flipped) crossing heuristic** achieve the best distribution fits both in terms of average performance as well as the variability across random seeds, indicating that this heuristic is able to more reliably capture the interactions between agents compared to the other architecture variants.

Table 4.4: RounD: average results across five random seeds for the three rounD locations. Per metric, the best performing model is underlined, while bold values highlight the best performing version of GMoP.

	Location 0			Location 1			Location 2		
Models	minADE	minFDE	NLL	minADE	minFDE	NLL	minADE	minFDE	NLL
ADAPT	9.76 ±0.38	16.15 ±0.60	$5.45e^3$ ±$3.10e^3$	7.23 ±0.46	12.69 ±0.95	$2.74e^3$ ±$1.02e^3$	6.97 ±0.27	12.77 ±0.86	$7.33e^3$ ±$1.00e^4$
AutoBots	5.38 ±0.50	13.30 ±1.24	$1.71e^4$ ±$4.76e^3$	3.14 ±0.01	8.51 ±0.09	$8.32e^3$ ±$1.08e^3$	2.95 ±0.18	7.73 ±0.50	$6.54e^3$ ±$1.05e^3$
FJMP	4.50 ±0.49	11.55 ±1.66	$6.46e^3$ ±$1.97e^3$	2.74 ±0.07	8.12 ±0.24	$7.75e^3$ ±571.68	3.22 ±0.37	9.26 ±0.71	$4.81e^3$ ±614.46
GMoP w/o H	7.23 ±2.54	16.36 ±4.27	212.01 ±399.98	4.30 ±1.96	11.49 ±5.54	−65.97 ±24.96	3.44 ±0.34	9.19 ±0.98	−53.17 ±15.80
GMoP indep	6.33 ±0.43	14.89 ±0.96	−52.83 ±15.31	3.59 ±0.23	9.52 ±0.51	−70.32 ±15.74	**3.37** ±0.13	**8.91** ±0.55	−89.47 ±9.36
GMoP euclH	5.91 ±0.23	13.84 ±0.55	67.07 ±64.59	**3.30** ±0.23	**8.68** ±0.63	−69.60 ±29.09	3.41 ±0.11	**8.91** ±0.33	−62.41 ±21.73
GMoP crH	5.89 ±0.46	13.78 ±0.97	82.36 ±144.12	3.48 ±0.15	9.14 ±0.46	−61.18 ±21.68	3.55 ±0.21	9.27 ±0.38	−43.32 ±29.72
GMoP hcrH	**5.67** ±0.30	**13.44** ±0.80	−3.79 ±12.16	3.54 ±0.24	9.29 ±0.56	−50.08 ±23.40	3.56 ±0.13	9.59 ±0.43	−51.14 ±13.38
GMoP flip crH	6.47 ±0.47	15.00 ±0.83	126.99 ±148.71	3.48 ±0.29	9.12 ±0.72	−73.36 ±17.32	3.55 ±0.28	9.22 ±0.62	−67.26 ±17.48
GMoP flip hcrH	6.16 ±0.27	14.69 ±0.66	−19.43 ±27.93	3.60 ±0.21	9.50 ±0.51	−55.40 ±22.48	3.40 ±0.11	8.96 ±0.30	−65.80 ±8.81

4.6 Conclusion & Discussion

Our findings indicate that relying purely on the network to extract interactions between agents based on the training data is often not sufficient for achieving good prediction performance. In fact, GMoP with no heuristics was generally the poorest performing architecture across our different test cases. At the same time, using the independence assumption and relying on the joint NLL loss to guide the learning of the network often lead to some of the better performing models. This is not to say, however, that heuristics are not beneficial for interpreting the interactions between agents. With the correct heuristics, a clear boost in performance can be achieved as shown on the NuScenes dataset, where the (flipped) crossing heuristic resulted in the best performance between the GMoP variants. What is further worth noting is that in the case of little training data, choosing a good heuristic which captures the underlying mechanism of interaction in the given scenario can aid in the learning of the model. We observed an example of this on RounD Location 0 when using the (flipped) hypothetical crossing heuristics as opposed to the (flipped) crossing heuristics. What is interesting, is that although the choice of heuristic can have a strong impact on the final performance of the network, the direction of the established edges in the graph does not greatly impact performance. We can see this in the comparable performance achieved by the (hypothetical) crossing heuristics compared to their flipped counterparts. This indicates that the presence of the edges is more important than the semantic interpretation of their direction. However, none of the heuristics provided the best performance across the different datasets, indicating that these heuristics are highly situation dependent. While the (flipped) crossing heuristics generally resulted in better performing models, it was in some cases outperformed by the (flipped) hypothetical crossing heuristics and independence assumption, such as on INTERACTION and RounD Location 0.

This study acts as a first step towards understanding how to best represent interactions between agents and leverage this for joint predictions. Nevertheless, there are aspects which our study does not investigate and would be important directions for future research. For one, we do not consider potential temporal evolution of the interactions themselves. While the (hypothetical) crossing heuristics strive to capture knowledge about the future evolution of the interaction between two agents it remains a simplification of the continuous negotiations and re-assessments of a situation that traffic participants go through with each other. Secondly, we make the assumption that the interactions between different agent types are governed by the same mechanisms. However, this need not be the case particularly when considering interactions between and within vulnerable and non-vulnerable road users. It would therefore be worthwhile to investigate whether there is a unified mechanism underlying all interactions, or whether interactions should be described differently depending on agents' types. Thirdly, we only look at the trajectories of the agents to determine their influence on each other and disregard potential confounders such as traffic rules which can have an important impact on the behavior of the agents. Lastly, while the type of heuristics we used in our study are common to the field of trajectory prediction they are fairly simplified representations of agent interactions. Meanwhile, the field of human behavior prediction has been striving towards modeling the underlying mechanisms behind the interaction between agents [117, 118]. These models, while grounded by human behavior studies, are often geared towards specific scenarios such as car following, intersections, merging, etc. and are often also not suitable for real-time multi-modal predictions which are relevant for motion planning. For this reason, another interesting direction for future research would be to investigate how to combine the benefits of scalability and real-time inference of neural networks with the deeper understanding of human interactions from the field of human behavior modeling.

5

Mode Collapse Happens: Evaluating Critical Interactions in Joint Trajectory Prediction Models

Autonomous Vehicle decisions rely on multimodal prediction models that account for multiple route options and the inherent uncertainty in human behavior. However, models can suffer from mode collapse, where only the most likely mode is predicted, posing significant safety risks. While existing methods employ various strategies to generate diverse predictions, they often overlook the diversity in interaction modes among agents. Additionally, traditional metrics for evaluating prediction models are dataset-dependent and do not evaluate inter-agent interactions quantitatively. To our knowledge, none of the existing metrics explicitly evaluate mode collapse.

In this chapter, we propose a novel evaluation framework that assesses mode collapse in joint trajectory predictions, focusing on safety-critical interactions. We introduce metrics for mode collapse, mode correctness, and coverage, emphasizing the sequential dimension of predictions. By testing four multi-agent trajectory prediction models, we demonstrate that mode collapse indeed happens. When looking at the sequential dimension, although prediction accuracy improves closer to interaction events, there are still cases where the models are unable to predict the correct interaction mode, even just before the interaction mode becomes inevitable.

This chapter is a verbatim copy of the pre-print [119]:

*M. Hugenholtz, **A. Mészáros**, J. Kober, Z. Ajanovic. "Mode Collapse Happens: Evaluating Critical Interactions in Joint Trajectory Prediction Models," under review.*

Statement of contributions: Maarten Hugenholtz contributed to the initial idea of the framework, developed the code base, and contributed to the writing of the paper. Anna Mészáros supervised the development of the framework and contributed to the writing of the paper. Jens Kober provided valuable feedback at all steps of the project. Zlatan Ajanovic provided valuable feedback at all steps of the project and supported the supervision.

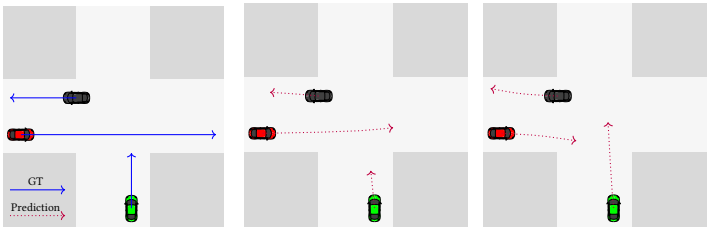
5.1 Introduction

Autonomous vehicles (AVs) have the potential to revolutionize personal transportation, motivated by improved driving comfort, energy efficiency and road safety [120]. Part of the autonomous driving challenge involves the planning of safe, comfortable and efficient driving trajectories in settings where potential future outcomes can be modeled as hybrid discrete-continuous systems [121, 122]. To achieve this, modular planning systems rely on a prediction module that predicts the motion of surrounding vehicles [123]. As human behavior is naturally uncertain and multimodal, it is unrealistic to predict a single trajectory for each agent, without knowing the agent's intent. Instead, one must consider the possible discrete behavior modes an agent might adopt – such as turning, accelerating, or changing lanes for example – along with their associated feasible continuous motions. Multimodal trajectory prediction (MTP), first introduced by Gupta et al. [60], reflects this hybrid viewpoint by generating multiple trajectory hypotheses per agent to cover different possible modes.

However, a common problem in multimodal trajectory prediction models is their susceptibility to mode collapse. This machine learning phenomenon occurs when the model fails to learn the true distribution of modes and only outputs the most likely mode, or multiple modes average out into a single, infeasible mode [61]. In a safety-critical application like autonomous driving, it is crucial that such failures are avoided, as incomplete or inaccurate predictions, that are used in a downstream planner, could result in collisions. Several works try to address the mode collapse issue either by using goal-conditioned prediction and a diverse set of goals [124–126], by using training objectives that incentivize diverse predictions [99, 127] or by promoting distributions with high entropy [128]. Generally, these works consider mode collapse on the environmental level by generating diverse predictions that cover various route options, but little attention has been given to ensuring diversity in the interaction modes among agent trajectories (e.g., in the spatiotemporal domain [129]).

Furthermore, none of the existing metrics and analysis methods explicitly evaluate mode collapse and provide measurable results. Evaluating vehicle trajectory prediction (VTP) models is challenging because real traffic behavior is governed by both continuous vehicle dynamics and discrete interaction outcomes (e.g., who yields at an intersection). VTP models are usually evaluated in open-loop, and their performance is primarily evaluated with distance-based metrics that assess the models' accuracy against the ground truth. While these metrics are an obvious choice and easy to compute, they heavily depend on the dataset, making it impossible to compare models from different datasets and interpret the results. In Figure 5.1, we illustrate that these distance-based metrics fail to effectively distinguish discrete interaction modes between agents and that averaging distance error results among multiple scenarios does not provide an insightful interpretation of results. Additionally, existing evaluation methods typically average errors over all agent pairs and time-steps, which mixes irrelevant cases (e.g., distant agents, lane-following vehicles) with genuinely safety-critical interactions. As a result, they fail to capture how well a model predicts interaction modes (e.g., the green vehicle lets the red vehicle pass first), which we argue is the most safety-critical aspect of driving in a closed-loop setting. These interaction outcomes form a discrete "mode" that tightly couples with the continuous vehicle dynamics. In this sense, traffic interaction is inherently a hybrid system: continuous trajectories evolve within a mode, while switching between modes (e.g., yielding vs. passing) produces qualitatively different future evolutions. A meaningful evaluation of prediction models must therefore assess both components of this hybrid structure. Furthermore, the temporal consistency of the predictions was found to be an important factor for the planner's performance in closed-loop simulation and is often neglected when evaluating prediction models [130].

In this work, we evaluate mode collapse at the interaction-mode level, treating trajectory prediction explicitly as a hybrid-systems problem. We analyze when and how predictive distributions lose critical discrete-mode information and examine the temporal evolution of these errors, which is particularly important in closed-loop settings. Our contributions are fourfold: First, we employ a homotopy-based representation to introduce an explicit metric for mode collapse based on mode



(a) Ground truth (b) Correct mode (c) Incorrect mode

Figure 5.1: We consider an exemplary intersection scenario, with two interacting vehicles, and one non-interacting vehicle. Both predictions (b) and (c) have similar mean final displacement errors, while only (b) correctly predicts the interaction mode between the green and red vehicle, as in the ground truth (a).

correctness and coverage. Second, we introduce sequential variants of these metrics, providing insights into the temporal evolution and consistency of the predictions. Third, we filter scenarios and consider only the relevant portions of path-crossing interactions, thereby reducing the evaluation dependency on datasets and improving the interpretability of the metrics. Finally, we benchmark two state-of-the-art trajectory prediction models, along with two additional baseline models, on the nuScenes dataset [77] and evaluate them using our novel metrics. Our results show that the models exhibit mode collapse and, in some cases, fail to correctly predict the interaction mode between agents, even just before the interaction mode becomes inevitable.

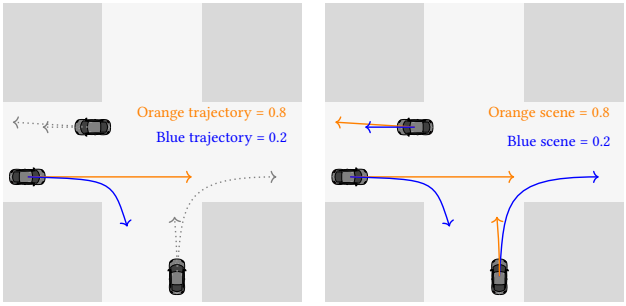
The rest of this paper is organized as follows: In Section 5.2, we give a brief literature overview on multimodal trajectory prediction models and the performance metrics used in popular benchmarks. In Section 5.4, we present our methodology and formulate our novel metrics. Section 5.5 describes the models that we tested and in Section 5.6 their performance on the nuScenes dataset is discussed, with both qualitative and quantitative results. Finally, Section 5.7 concludes this work, and we discuss limitations as well as exciting directions for future research in this area.

5.2 Related Works

In Section 5.2.1 we discuss how multimodal trajectory prediction models mitigate mode collapse, what mode representations have been used, and the difference between marginal and joint prediction. Section 5.2.2 discusses the current trajectory prediction evaluation frameworks, and how they fail to effectively evaluate interactions.

5.2.1 Multimodal trajectory prediction models

Multimodal trajectory prediction models employ various techniques to mitigate mode collapse. A common approach is to first enumerate diverse possible modes and then condition the prediction upon these modes, to enforce diverse predictions. A mode is an abstraction of a trajectory referring to a high-level behavior, and can be represented on the environment level (goal lanes or points) [124], vehicle level (lane change, accelerating, braking) [131] or interaction level (yielding, going) [132]. Using such modes as an intermediate representation to condition the prediction upon, improves interpretability and helps mitigate mode collapse. However, since no unified definition of a mode exists, there are also no metrics to quantify the discrete mode prediction performance of the models. In this work, we will focus on the interaction modes between agents, and use the concept of free-end



(a) Marginal prediction (b) Joint prediction

Figure 5.2: Illustration demonstrating the difference between marginal prediction (a) and joint prediction (b).

homotopy [133] to categorize them into clockwise and counterclockwise rotations. Multimodal trajectory prediction models can be categorized into node-centric and scene-centric models, which perform the prediction per-agent and jointly for the whole scene, respectively. Figure 5.2 demonstrates the difference. Generally, scene-centric models better capture the interactions among agents, have higher scene consistency and are more compatible with downstream planners [130]. On the other hand, node-centric models are easier to train and better cover the agents' motion [91]. In order to evaluate the interaction mode of a trajectory pair, joint trajectory predictions are required. Therefore, we will focus on the evaluation of scene-centric vehicle trajectory prediction models only. Categorical Traffic Transformer (CTT) [126] is an example of such a model. It uses an interpretable set of Scene Modes (SM) to supervise the latent mode distribution. Uniquely, these modes consist of two types: agent2lane (a2l) modes and agent2agent (a2a) modes, thereby capturing both the route and interaction intention of agents.

5.2.2 Trajectory prediction performance metrics

Vehicle trajectory prediction models are evaluated in open-loop, using various metrics that assess the accuracy, likelihood, diversity, and admissibility of the predicted trajectories. Distance-based metrics like the minimum average displacement error (minADE), minimum final displacement error (minFDE) and miss rate (MR) have been the primary accuracy metrics used to compare trajectory prediction models. However, the performance on these metrics is heavily dependent on the used dataset, making comparison between different datasets impossible and hindering the interpretation. Additionally, these metrics provide no insight into the predicted modes of the distribution.

Another aspect that has been neglected in the evaluation is the interactions among agents. Recent works [63, 99, 126] have turned to scene-centric models, to better capture interactions between agents by simultaneously rolling-out their future trajectories. The Waymo Open Motion Dataset (WOMD) [134] prediction benchmark introduced joint metrics for the minADE, minFDE and MR. Their definitions are similar to their marginal variants, except that the minimum error of K predictions is taken over the whole scene instead of agent-wise. This means that we cannot mix-and-match the best prediction for each agent over different scene samples, which means the prediction task is inherently more challenging but also gives a more realistic idea of the performance. While these joint metrics implicitly evaluate agent interactions, the lack of an explicit metric makes interpretation challenging, as demonstrated in Figure 5.1.

In CTT [126] - already introduced above - a2l and a2a modes are defined and used to condition the prediction task upon the scene mode. Additionally, they introduce corresponding mode metrics: the mode correct rate and mode cover rate. The mode correct rate is the percentage of most likely (ML) predictions that match the ground truth (GT) mode (a2a, a2l or both). The mode cover rate is the rate at which one of the K predicted trajectories matches the GT mode. They compare their performance on these metrics to AgentFormer (AF) [63] on the nuScenes and WOMD datasets. While this is a promising step towards formalizing modes and improving intention prediction (lane and interaction modes), their metrics lack interpretability and are still heavily dependent on the dataset. The latter is demonstrated by the fact that for AF there is almost a 50% performance difference in the a2a cover rate between nuScenes and the WOMD. In this work, we build on their mode metrics for a2a interactions, adapt the criteria and extend it to evaluate only critical scenarios and only relevant segments of those scenarios. Additionally, we evaluate the temporal evolution of the predictions over the scene duration. All this allows us to use metrics to quantify a model's interaction prediction performance in a more insightful and data-independent manner.

5.3 Problem Formulation

Standard trajectory metrics treat predictions as purely continuous signals, but hybrid behavior requires reasoning over both the discrete interaction mode and its induced continuous motion. Without explicitly capturing this mode structure, evaluation cannot distinguish between geometrically similar trajectories that encode fundamentally different interaction intentions.

To evaluate these interaction outcomes properly, we require: (1) a way to characterize discrete interaction modes, (2) a means to relate those modes to continuous trajectories, and (3) a formulation that isolates only the relevant time intervals and agent pairs.

5.3.1 Trajectories and Predictions

We consider two agents A and B with trajectories

$$\begin{aligned} \tau^{\mathrm{A}} &= \left[(x_1^{\mathrm{A}}, y_1^{\mathrm{A}}), \ldots, (x_N^{\mathrm{A}}, y_N^{\mathrm{A}})\right], \\ \tau^{\mathrm{B}} &= \left[(x_1^{\mathrm{B}}, y_1^{\mathrm{B}}), \ldots, (x_N^{\mathrm{B}}, y_N^{\mathrm{B}})\right], \end{aligned} \tag{5.1}$$

where (x_t, y_t) denotes the position of an agent at time frame t. Both trajectories are defined over the maximal interval $t \in [1, N]$ in which the two agents are simultaneously present in the scene.

At each time frame t, a trajectory prediction model $\mathcal{M}$ produces K joint multimodal predictions for the future horizon T_{pred}. We denote these predictions as

$$\begin{aligned} \mathbf{y}_{\text{pred,t}}^{\text{A,B}} &= \{(\widehat{\tau}_t^{\mathrm{A}}, \widehat{\tau}_t^{\mathrm{B}})_1, \ldots, (\widehat{\tau}_t^{\mathrm{A}}, \widehat{\tau}_t^{\mathrm{B}})_K\}, \\ \text{where} \quad \widehat{\tau}_t &= [(\widehat{x}_t, \widehat{y}_t), \ldots, (\widehat{x}_{t+T_{\text{pred}}}, \widehat{y}_{t+T_{\text{pred}}})]. \end{aligned} \tag{5.2}$$

Here, t denotes the current time from which predictions are generated, and should not be confused with the internal trajectory indices i defining future positions.

For evaluation, each set of predictions is compared with the corresponding ground-truth trajectory segments,

$$\begin{aligned} \mathbf{y}_{\text{gt,t}}^{A} &= \tau_{t:t+T_{\text{pred}}}^{A} \\ &= \left[(x_t^{\mathrm{A}}, y_t^{\mathrm{A}}), \ldots, (x_{t+T_{\text{pred}}}^{\mathrm{A}}, y_{t+T_{\text{pred}}}^{\mathrm{A}})\right] \\ \mathbf{y}_{\text{gt,t}}^{B} &= \tau_{t:t+T_{\text{pred}}}^{B} \\ &= \left[(x_t^{\mathrm{B}}, y_t^{\mathrm{B}}), \ldots, (x_{t+T_{\text{pred}}}^{\mathrm{B}}, y_{t+T_{\text{pred}}}^{\mathrm{B}})\right] \end{aligned} \tag{5.3}$$

However, continuous trajectories alone do not reveal which interaction outcome they correspond to. To evaluate VTP models at the level of interaction intentions, we must first identify which agent pairs actually participate in a discrete interaction and isolate the time intervals where mode decisions matter.

5.3.2 Interactions

Not all agent pairs form meaningful interactions. Many pairs (e.g., lane-following, distant vehicles, overtaking on separate lanes) cannot collide and should not contribute to interaction evaluation. Following [135], we define an inter-vehicle interaction as a situation in which the behavior of at least two road users is influenced by the possibility that both intend to occupy the same region of space at the same time in the near future.

In practice, many theoretically possible interactions rarely occur because vehicle motion is strongly constrained by infrastructure and traffic rules. Pairs of agents that remain on different lanes throughout the scene (e.g., passing traffic), or that travel on the same lane in a car-following configuration, do not meaningfully influence one another. The interactions of interest in this work are the *safety-critical path-crossing cases* in which two agents initially occupy different lanes but move toward a region of shared space, such as at merges or unsignalized intersections. Figure 5.3 illustrates typical interacting and non-interacting examples.

5.3.3 Interaction Modes via Free-End Homotopy

To represent the discrete component of this hybrid interaction system, we use a topological encoding of trajectory relationships. The key idea is that different interaction outcomes correspond to different homotopy classes. Homotopy groups trajectories into equivalence classes according to whether one trajectory can be continuously deformed into another without intersecting obstacles [136]. Free-end homotopy removes the requirement that trajectories share endpoints, which is essential when comparing multiple model predictions with ground-truth trajectories that differ in their future intentions [133]. In this setting, an interaction mode can be determined by the relative rotation of one agent around the other, i.e., the winding angle.

Definition 1 (Winding angle) *The winding angle represents the cumulative angular change between the relative positions of agents A and B along their trajectories. For each time frame t, let*

$$\alpha_t = \arctan\left(\frac{y_t^{\mathrm{A}} - y_t^{\mathrm{B}}}{x_t^{\mathrm{A}} - x_t^{\mathrm{B}}}\right)$$

denote the bearing angle of A with respect to B. The winding angle between the two trajectories is then defined as

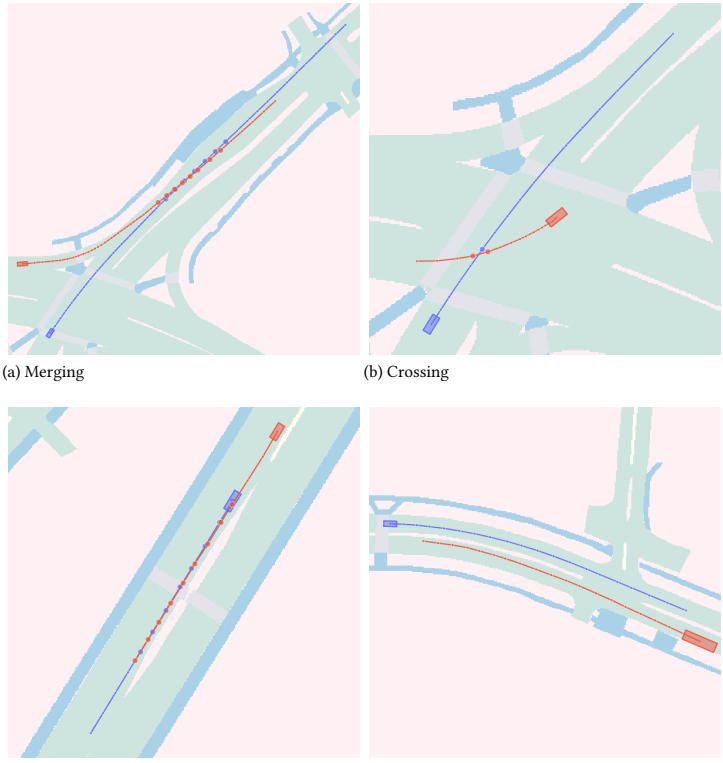
$$\Delta\theta(\tau^A, \tau^B) = \sum_{t=1}^{N-1} \left(\alpha_{t+1} - \alpha_t\right), \tag{5.4}$$

i.e., the accumulated angular difference over time.

This angle captures how one agent rotates around the other over time. Figure 5.4 shows two typical interaction patterns: clockwise (CW) when one agent yields, and counterclockwise (CCW) when the other yields.

Definition 2 (Free-end homotopy) *A pair of trajectories* (τ^A, τ^B) *is assigned a* free-end homotopy class *based on the winding angle* $\Delta\theta(\tau^A, \tau^B)$*. The interaction mode is defined as*

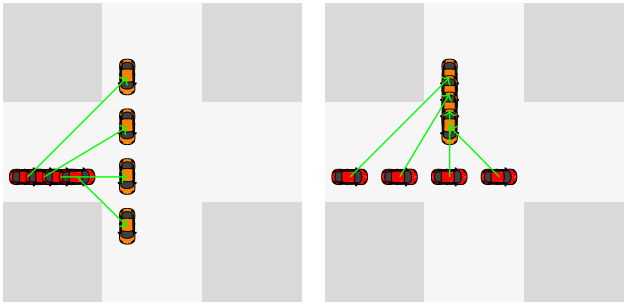
$$\mathrm{h} := \begin{cases} CW, & \Delta\theta\left(\tau^{\mathrm{A}}, \tau^{\mathrm{B}}\right) < -\hat{\theta} \\ S, & -\hat{\theta} \le \Delta\theta\left(\tau^{\mathrm{A}}, \tau^{\mathrm{B}}\right) < \hat{\theta} \\ CCW, & \Delta\theta\left(\tau^{\mathrm{A}}, \tau^{\mathrm{B}}\right) \ge \hat{\theta}, \end{cases} \tag{5.5}$$



(a) Merging (b) Crossing

(c) Car-following (d) Passing vehicles

Figure 5.3: Exemplary traffic scenarios of safety-critical interaction agent-pairs (a, b) and non-interacting agent-pairs (c, d) from the nuScenes dataset [77]. The time-steps where the agents are on the commonly shared path are visualized with big markers.



(a) CW class (b) CCW class

Figure 5.4: Visualization of angular distance calculation and convergence of two agents traversing an intersection. The figure shows the cars and their relative angles at four consecutive time-steps. In (a) the red vehicle yields, resulting in a clockwise rotation, and in (b) the orange vehicle yields, resulting in a counterclockwise rotation.

where $\hat{\theta}$ is a small angular threshold separating static interactions from clockwise and counterclockwise interaction modes $[S, CW, CCW]$.

The CW and CCW classes are visualized in Figure 5.4. We omit the static class because, in safety-critical path-crossing interactions, the relative rotation between agents is non-negligible and the static case does not occur in practice.

Together, these components enable an evaluation that respects the hybrid nature of traffic interactions, assessing both the discrete mode prediction and its consistency with the continuous trajectory evolution.

5.3.4 Problem Statement

Given a scene with a set of agent pairs $\mathcal{A}$, their ground-truth trajectories, and a trajectory prediction model $\mathcal{M}$ that outputs K joint multimodal predictions per time frame, our objective is to evaluate how accurately, completely, and consistently the model predicts the discrete interaction mode (CW or CCW) during safety-critical path-crossing interactions.

We aim to formulate an evaluation procedure that identifies the relevant agent pairs and time intervals, assigns homotopy-based interaction modes to predicted and ground-truth trajectories, determines at which time steps multiple interaction outcomes remain physically feasible, and measures the resulting prediction performance in terms of mode correctness, mode coverage, mode collapse, and temporal consistency. This formulation provides the foundation for the methodology described in Section 5.4.

5.4 Methodology

In this section, we introduce an evaluation framework that explicitly targets the hybrid nature of interaction prediction, rather than averaging errors over all agents and time-steps. Our approach proceeds in three stages. First, we filter datasets to retain only safety-critical path-crossing interaction scenes (Section 5.4.1). Second, for each interacting pair, we use a two-class free-end homotopy representation and dynamically simulated roll-outs to enumerate feasible interaction modes and determine the interval over which the multiple outcomes are still feasible (Section 5.4.2, Section 5.4.3).

Third, within this interval, we evaluate the model's interaction-mode predictions using metrics for correctness, coverage, mode collapse, and temporal consistency (Section 5.4.4). The overall procedure is summarized in Algorithm 2.

Algorithm 2 Interaction Evaluation

Require: $\mathcal{A}$: set of agent pairs
for each (A, B) $\in \mathcal{A}$ **do**
  if ¬IsSafetyCritical(A, B) **then** ▷ *Sec 5.4.1*
    continue
  ▷ *loop over common time frames*
  for each $t \in [1, N]_{A,B}$ **do** ▷ *calculated from t,.. t+Tpred*
    $\mathbf{y}_{\text{pred,t}} \leftarrow$ GetModelPred(A, B, t) ▷ *Eq 5.2*
    $h_{\text{pred,t}} \leftarrow$ GetHomotopy$(\mathbf{y}_{\text{pred,t}})$
    $\mathbf{y}_{\text{gt,t}} \leftarrow$ GetGroundTruth(A, B, t) ▷ *Eq 5.3*
    $h_{gt,t} \leftarrow$ GetHomotopy$(\mathbf{y}_{\text{gt,t}})$

    ▷ *Sec 5.4.3*
    $\mathbf{y}_{\text{roll,t}} \leftarrow$ SimRollOuts(A, B, t)
    $\mathbf{y}_{\text{feas,t}} \leftarrow$ CheckFeasibility$(\mathbf{y}_{\text{roll,t}})$
    $h_{\text{feas,t}} \leftarrow$ GetHomotopy$(\mathbf{y}_{\text{feas,t}})$
    if (CW $\notin h_{\text{feas,t}}$) **or** (CCW $\notin h_{\text{feas,t}}$) **then**
      ▷ *Inevitable homotopy state (Def 5)*
      break

    ▷ *Sec 5.4.4*
    EvalMetrics$(h_{\text{pred,t}}, h_{\text{gt,t}})$

5.4.1 Filtering safety-critical, interactive scenarios

In Figure 5.3 we saw that not every pair of agents gives rise to a meaningful interaction. Merging and crossing scenarios are of primary interest: the agents initially travel on different lanes but later enter a shared lane segment, creating a potential collision conflict. To automatically identify such safety-critical interactions we detect for each agent the first time at which its position overlaps with the other agent's trajectory. We refer to these time frames as the path-sharing time frames t^A_{PS} and t^B_{PS}. Formally, these two time frames are obtained as:

$$\begin{aligned} t^A_{\text{PS}} &= \min t \ \ni \min \left\| \tau^B - \tau^A_{0,t} \right\| < \epsilon \\ t^B_{\text{PS}} &= \min t \ \ni \min \left\| \tau^A - \tau^B_{0,t} \right\| < \epsilon \end{aligned} \tag{5.6}$$

where $\tau_{0,t} = [(x_t, y_t) \ldots (x_t, y_t)]$ is an agent's static trajectory at time frame t, and ϵ is a distance threshold. A safety-critical interaction is thus defined as follows.

Definition 3 (Safety-critical interactions) *We define an interaction as safety-critical if both of the following criteria are fulfilled:*

1. *The agents' paths cross some time after the beginning of the scene, i.e.* $\exists\, t^A_{\text{PS}}, t^B_{\text{PS}} \in (1, N]$

2. *The time-difference between both agents entering the common path is small (i.e., we are interested in closely interacting vehicles):* $\Delta T_{\text{PS}} = \left|t^A_{\text{PS}} - t^B_{\text{PS}}\right| \leq T_{\text{crit}}$.

With the above definitions, we can filter the safety-critical interactions, like merging and crossing, from basic car-following and traffic light scenarios. This reduces the dependency on the dataset as we only evaluate similar and safety-critical interactions. In traditional trajectory prediction evaluation all cars and scenes are considered, which complicates interpretation because the distance errors are averaged, making it unclear what kind of scenarios were evaluated and how the model performed in critical cases. Thus, by applying our methodology, the metrics become more interpretable and insightful.

5.4.2 Categorizing interaction modes using homotopy

To categorize interactions between agent-pairs we use the concept of free-end homotopy, as defined in definition 1 and definition 2. However, in contrast to [126, 133], we set the threshold $\hat{\theta}$ to zero, effectively eliminating the static class in Equation (5.5). A fixed non-zero threshold can lead to ambiguities, as the angular distance $\Delta\theta$ not only depends on the speed and intention of the agents, but also on the road topology and the prediction horizon. By eliminating the static class, we always have a distinct interaction class for a trajectory pair. This is especially important for the predictions, as they might not be close to the ground truth, but still contain the model's implicit homotopy class prediction, i.e., the intuition for how the agents will interact (CW or CCW rotation with respect to each other). Besides, since we only evaluate the safety-critical path-crossing interactions (Section 5.4.1), most static interactions like car-following and distant agent-pairs will already be filtered out, making the static class redundant.

Besides filtering interaction scenarios for evaluation, we also want to segment the interesting temporal duration of those scenarios, i.e., once an interaction has happened, there is no point in further evaluating it. E.g. Figure 5.4 depicts the process of calculating the angular distance over time for two agents traversing an intersection. From this figure, it becomes clear that the angular distance is only significant if the agents are close. Furthermore, once either of the agents has entered the shared path (the middle of the intersection in this case), the homotopy class of the interaction is inevitable and the angular distance converges afterward. Geometrically, this occurs at $t_{\text{IHS,kin}} = \min\{t^A_{PS}, t^B_{PS}\}$. However, it would be too imprecise to use this as a criterion, as the real moment is earlier because vehicles cannot instantly accelerate and decelerate, and are larger than a singular point in space. Thus, to find the true instance at which the homotopy class becomes inevitable dynamic simulations are needed, which will be discussed in the next subsection.

5.4.3 Enumerating feasible homotopy classes

We enumerate feasible homotopy classes by simulating alternative interaction outcomes for agent-pairs. At each time-frame t in the scene, we accelerate one agent and decelerate the other, and vice versa. Thus, the set of future roll-outs for the agent-pair (A,B) at time-frame t is:

$$\mathbf{y}_{\text{roll,t}} = [(\tau^{\text{A}}_{\text{decel,t}}, \tau^{\text{B}}_{\text{accel,t}}), (\tau^{\text{A}}_{\text{accel,t}}, \tau^{\text{B}}_{\text{decel,t}})]. \tag{5.7}$$

We keep the ground truth paths of the agents, and only alter the velocity profile of the agents (either accelerate or decelerate), whilst keeping both longitudinal and lateral accelerations within realistic limits for comfortable driving. Since we only simulate roll-outs for safety-critical interaction pairs, the interaction class is CW for one roll-out and CCW for the other. By keeping the ground-truth paths of the agents the same, we ensure that the simulations are realistic (i.e., they are based on the ground-truth, so they comply with the road topology) and can be computed efficiently.

To check that the agents do not collide in either roll-out, we use a binary collision detection function, denoted by IsCollision$(\tau^{\text{A}}, \tau^{\text{B}})$. To take the vehicle dimensions and headings into account,

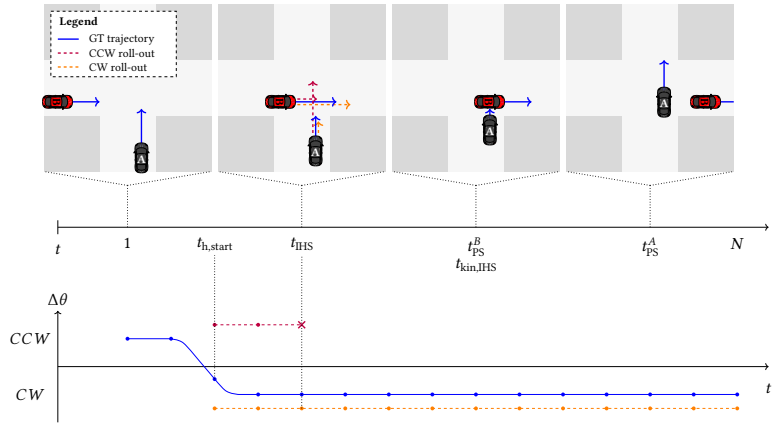


Figure 5.5: Important Timesteps for the Evaluation Framework, with visualizations of the vehicle trajectories including the simulated roll-outs at t_{IHS} (above), and the winding angle of the trajectories and feasible simulated roll-outs (below). Note that at t_{IHS}, the CCW homotopy class stops being feasible as the roll-out for this class results in a collision.

we take inspiration from [137], and fit three disks with radii $r^{\text{A}} = \frac{1}{2}$width to each vehicle: at the vehicle center and at both bumpers. A collision is detected by computing the minimum distances between all disks of both vehicles, for all time-steps i of a roll-out. The vehicles are in collision if the minimum distance d_{min} is smaller than the sum of the disk radii r fitted to the vehicles: $d_{\text{min}} < r^{\text{A}} + r^{\text{B}} = d_{\text{collision}}$. Whilst there can still be hypothetical cases where a collision is missed, this approach works in most practical cases and is computationally efficient. Through this we obtain the feasible roll-outs.

Definition 4 (Feasible homotopy classes) *Feasible roll-outs are characterized by:*

$$\mathbf{y}_{\text{feas,t}} = \{(\tau^{\text{A}}, \tau^{\text{B}}) \in \mathbf{y}_{\text{roll,t}} \mid \neg \text{IsCollision}(\tau^{\text{A}}, \tau^{\text{B}})\}. \tag{5.8}$$

And consequently, the unique set of feasible homotopy classes is:

$$\mathbf{h}_{\text{feas,t}} = \{h(\tau^{\text{A}}, \tau^{\text{B}}) \mid (\tau^{\text{A}}, \tau^{\text{B}}) \in \mathbf{y}_{\text{feas,t}}\}, \tag{5.9}$$

where $\mathbf{h}_{\text{feas,t}} \in \{CW, CCW\}$.

From this we can determine the ground truth inevitable homotopy state (IHS) as follows.

Definition 5 (Inevitable homotopy state) *The final ground truth homotopy class* $h_{\text{gt,final}}$ *represents the true outcome of the interaction. We define the Inevitable Homotopy State (IHS) as the time frame at which the corresponding interaction class becomes inevitable, i.e., only one unique homotopy class is still feasible (non-colliding). Formally:*

$$t_{\text{IHS}} = \min\{\, t \mid |\mathbf{h}_{\text{feas,t}}| = 1 \,\}. \tag{5.10}$$

This point in time, as well as the other important time steps of our methodology, are visualized in Figure 5.5.

5.4.4 Evaluation interarval

In accordance to the above definitions, we evaluate the predictions of a VTP model until the last frame at which both classes are still feasible, i.e.:

$$t_{\text{h,final}} = \max\{\ t \mid |\mathbf{h}_{\text{feas,t}}| = 2\ \}. \tag{5.11}$$

The evaluation starts once the homotopy class starts to converge towards the inevitable homotopy state, but at most a whole prediction horizon T_{pred} before then:

$$\begin{aligned} &t_{\text{h,start}} = \max\{\ \ t \mid\ h_{\text{gt,t}} = h_{\text{gt,final}} \wedge h_{\text{gt,t}-1} \neq h_{\text{gt,final}}\ \} \\ &\text{for} \quad t \in [t_{\text{h,final}} - T_{\text{pred}}, t_{\text{h,final}}]. \end{aligned} \tag{5.12}$$

Thus the evaluation interval is $[t_{\text{h,start}}, t_{\text{h,final}}]$. Note that the duration of this interval varies, and in many cases is shorter than T_{pred}, because the interval for which both agents are recorded in the data is shorter or the ground truth homotopy class starts to converge later.

5.4.5 Evaluating interaction mode prediction performance

Since we want to evaluate the interaction between trajectories of agent-pairs, we require joint multi-agent predictions, in the form of those provided in Equation (5.2). The set of homotopy classes of the model's predictions is:

$$\mathbf{h}_{\text{pred,t}} = \{h((\widehat{\tau}_t^{\text{A}}, \widehat{\tau}_t^{\text{B}})_{\text{k}}) \mid (\widehat{\tau}_t^{\text{A}}, \widehat{\tau}_t^{\text{B}})_{\text{k}} \in \mathbf{y}_{\text{pred}}\}. \tag{5.13}$$

To evaluate the model's ability to correctly predict the interaction mode, we follow [126] in defining mode correctness and coverage.

Definition 6 (Mode correctness) *The a2a mode is correct if the most likely (ML) prediction's mode $h_{\text{ml,t}}$ corresponds to the ground truth mode $h_{\text{gt,t}}$, i.e., $h_{\text{ml,t}} = h_{\text{gt,t}}$.*

Definition 7 (Mode coverage) *The a2a mode is covered if one of the K predictions covers the ground truth mode, i.e., $h_{\text{gt,t}} \in \mathbf{h}_{\text{pred,t}}$.*

To get insight into the temporal evolution of the interaction prediction, we propose a time-based metric: the time-to-correct-mode-prediction ($\Delta T_{\text{correct}}$), which is the time the model needs to recognize the intention of the cars before the interaction has settled, i.e., before the inevitable homotopy state is reached.

$$\begin{aligned} t_{\text{incorrect}} &= \max\{t \mid h_{\text{gt,t}} \neq h_{\text{ml,t}}\} \\ \Delta T_{\text{correct}} &= t_{\text{h,final}} - t_{\text{incorrect}} \end{aligned} \tag{5.14}$$

Similarly, we compute the time-to-covered-mode-prediction ($\Delta T_{\text{covered}}$). The difference is that we consider all K predictions of the model, instead of just the most likely one.

$$\begin{aligned} t_{\text{uncovered}} &= \max\{t \mid h_{\text{gt,t}} \notin \mathbf{h}_{\text{pred,t}}\} \\ \Delta T_{\text{covered}} &= t_{\text{h,final}} - t_{\text{uncovered}} \end{aligned} \tag{5.15}$$

If the predictions are correct or covered from the beginning of the prediction interval, we cannot calculate the respective times, because we cannot make any assumptions about the model's predictions before then. In these cases, we consider the predictions a discrete correct class rather than a time. Therefore, we report three metrics, aggregated over all interactions: the mean times before the predictions have the correct mode or cover it, the percentage of predictions that are correct or covered from the beginning of the evaluation interval ($@t_{\text{h,start}}$), and the percentage of predictions where $\Delta T = 0$, meaning the predictions are wrong even just before the inevitable homotopy state ($@0s$).

Definition 8 (Mode collapse) *We define a2a mode collapse an interaction mode being feasible, but not predicted by any of the model's predictions, i.e.,* $\mathbf{h}_{\text{feas,t}} \not\subseteq \mathbf{h}_{\text{pred,t}}$.

So, mode collapse does not necessarily consider the ground truth, but the feasibility of hypothetical future roll-outs. Finally, we define the mode collapse rate as the percentage of time-steps in the relevant interval $t \in [t_{\text{h,start}}, t_{\text{h,final}}]$ where mode collapse occurs. It is worth noting that in many cases (i.e., scenes with many agents) it is impossible for the model to cover all feasible modes with a finite number of joint predictions, due to the cardinality of the mode space growing exponentially with the number of agents.

5.4.6 Temporal consistency of predictions

For safe motion planning, it is desirable that a model's interaction mode predictions evolve smoothly over time. Small changes in observed motion between consecutive frames should not trigger unnecessary switches in the predicted interaction mode. To assess this aspect, we evaluate the temporal consistency of the model's most likely (ML) mode predictions.

Definition 9 (Mode prediction consistency) *For an agent pair, the ML mode predictions are said to be* consistent *over the evaluation interval* $t \in [t_{\text{h,start}}, t_{\text{h,final}}]$ *if the predicted mode changes at most once. Otherwise, the predictions are deemed inconsistent.*

The mode prediction consistency is a hit-or-miss metric. For example, a sequence of predictions [*CW, CCW, CCW*] is considered consistent, as a single correction is allowed, whereas [*CCW, CW, CCW*] is inconsistent due to multiple mode switches.

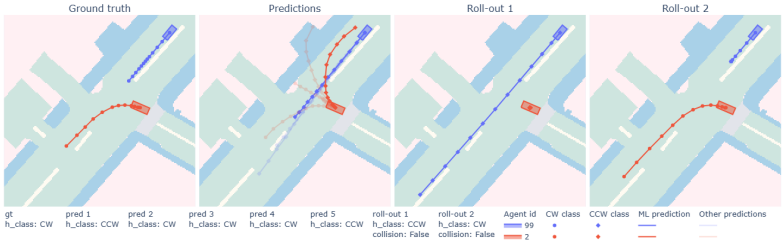


Figure 5.6: Visualization of the interaction mode evaluation for AgentFormer on agent-pair (99,2) at frame 11 in scene-0103 of the nuScenes dataset. For the predictions, only the ML prediction is shown with full opacity. The corresponding homotopy classes (h_class) are shown in the legend, and displayed in the plot using ◦ and ⋄ markers for the CW and CCW class, respectively. Additionally, the collision status is shown in the legend for both roll-outs. For this specific frame, both interaction modes are still feasible. The mode is not predicted correctly (ML prediction), but it is covered by one of the other predictions.

5.4.7 Implementation example

Let us look at an example from AgentFormer's (AF) [63] predictions on one of the validation scenes of the nuScenes dataset [77]. In this scene, only the relevant interacting agent-pairs are considered. At each frame, we simulate future roll-outs and check their feasibility with the collision checker. Furthermore, we compute the homotopy classes of the ground truth, the predictions and the roll-outs. In Figure 5.6, we visualize this process for a single frame. Table 5.1 shows an overview of the interaction modes of predictions and roll-outs for all relevant frames, i.e., falling within the evaluation interval, of this interaction-pair.

Table 5.1: Example mode metrics evolution for AF's predictions on agent-pair (99,2) in scene-0103 of the nuScenes dataset. The ground truth mode is covered from the beginning, but only predicted consistently correct from frame 13 onwards. Thus, the predictions are inconsistent in this case.

frame	GT mode	ML mode	all K modes	feasible modes	mode correct	mode covered	mode collapse
5	CW	CW	CW	CCW, CW	✓	✓	✓
6	CW	CW	CW	CCW, CW	✓	✓	✓
7	CW	CW	CW	CCW, CW	✓	✓	✓
8	CW	CW	CW	CCW, CW	✓	✓	✓
9	CW	CW	CW	CCW, CW	✓	✓	✓
10	CW	CW	CW	CCW, CW	✓	✓	✓
11	CW	CCW	CCW CW	CCW, CW		✓	
12	CW	CCW	CCW CW	CCW, CW		✓	
13	CW	CW	CW	CCW, CW	✓	✓	✓
14	CW	CW	CW	CCW, CW	✓	✓	✓
15	CW	CW	CW	CCW, CW	✓	✓	✓
16	CW	CW	CW	CW	✓	✓	

For this specific interaction, the inevitable homotopy state is at frame 16, as only the CW mode is still feasible from then, as the CCW mode would end in a collision. We wish to evaluate the mode predictions for a whole prediction horizon T_p before then. However, in many cases (such as this example), this is not possible, simply because the interval for which both agents are recorded in the dataset is not long enough. Thus, we will evaluate the mode predictions from the first point at which there are predictions for both agents, until the inevitable homotopy state. In this case: from frame 5 until frame 15. Since nuScenes is recorded at 2Hz, we find that it takes the model $\Delta T_{\text{correct}} = 1.5$s to correctly predict the interaction class. For $\Delta T_{\text{covered}}$ we see that the predictions cover the ground truth class from the start of the evaluation interval. Since there are no prior time-steps available, we consider such cases a correct/covered class, rather than a time. Furthermore, from the table, it becomes clear that the predictions are inconsistent because the ML prediction's mode changes more than once. Finally, in 9 out of the 11 frames the model did not predict all feasible modes, so the mode collapse rate for this scene is 81.8%.

5.5 Trajectory Prediction Models

We test our novel evaluation methodology on the nuScenes dataset [77] and report results for four models: AgentFormer (Section 5.5.1), Categorical Traffic Transformer (Section 5.5.2), a constant velocity model (Section 5.5.3) and an oracle model (Section 5.5.4). In the following subsections, we briefly discuss the characteristics and implementation of these models.

5.5.1 AgentFormer

AgentFormer (AF) [63] is a multi-agent trajectory prediction model. They utilize a transformer-based architecture, that simultaneously models the social and temporal dimension of agents. Their prediction framework jointly models the agents' intentions, to predict diverse and socially-aware future trajectories [138]. We will utilize their pre-trained nuScenes model, and use the version which outputs $K = 5$ multi-agent trajectories.

5.5.2 Categorical Traffic Transformer

Categorical Traffic Transformer (CTT) [126] is a multi-agent trajectory prediction model, with an interpretable latent space consisting of agent-to-agent and agent-to-lane modes. CTT generates diverse behaviors by conditioning the trajectory prediction on different modes. The authors published their code including pre-trained weights for the nuScenes dataset [139]. Unfortunately, we did not succeed in reproducing the numbers reported in their paper and uncovered various issues, making direct comparison with the other models difficult. Firstly, their pre-trained model is trained for a prediction horizon of 3 seconds, whereas AF is trained for 6 seconds, as dictated by the nuScenes benchmark [77]. To match the varying prediction horizons, the 6-second predictions from AF are cut to 3 seconds. Secondly, all K predicted modes and trajectories are identical, making the model effectively unimodal. Finally, whereas AF predicts for all vehicles in the scenes, CTT predicts only for the road users within a certain attention radius of the ego-vehicle, but it does include pedestrians whereas AF does not. We use AF's data preprocessing backbone and match CTT's predictions to the corresponding agents. However, due to the aforementioned attention radius used in CTT, many predictions are missing for certain agents. In these cases, the current ground truth position is kept static and used as a prediction instead. Due to these issues, we are not able to report the real performance of CTT on interaction prediction. However, we still report the metrics and compare them to the other models, to set a baseline and show that our methodology generalizes to other models.

5.5.3 Constant velocity model

The constant velocity (CV) model is a simplistic unimodal model that assumes the vehicle will remain in its current heading and velocity [140]. Because it produces a single mode, it inherently suffers from mode collapse. However, it is an interesting baseline for comparison, because it tells us in how many scenarios we can correctly assess the vehicle pair's interaction class by simply extrapolating their current trajectories.

5.5.4 Oracle model

As the cardinality of the space of interaction modes grows exponentially with the number of agents in the scene, and trajectory prediction models typically predict a fixed set of K modes, covering all feasible modes becomes infeasible in scenes with many agents. To test this limitation, we propose a multimodal oracle model. The oracle's goal is to predict a set of K multimodal trajectories that cover all feasible modes of the interacting agents. The oracle will be given access to the agents' ground truth paths, so it knows which agents will be interacting, i.e., crossing the same path, in the near future. However, the trajectories are unknown, i.e., it does not know the velocity profiles along the path, so the interaction class is still to be determined by the model. The oracle's goal is to cover all feasible interaction modes between the path-crossing agent-pairs. Analogously to the methodology described in Section 5.4.3, we keep the agents' ground truth paths and simulate future roll-outs with a constant velocity, deceleration, or acceleration profile. Firstly, all agents are initialized with their constant velocity profile. Next, we determine all combinations of constant velocity, acceleration, and deceleration profiles between the interacting agents and reject the combinations with collisions. Finally, we must assign each joint prediction a likelihood. We argue that the likelihood of a joint scene prediction is proportional to the overall utility in the scene, where the average speed of a roll-out combination can be used as a utility measure. Therefore, to get a finite set of K joint predictions, we compute the average speed of the roll-outs and output the top-K trajectory combinations with the highest average speed.

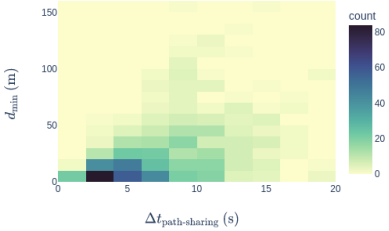


Figure 5.7: Density heatmap of the path-sharing interactions in the full train-validation split of nuScenes. The interactions are characterized in closeness, with the real-time closest distance on the y-axis and the time difference between the agents occupying the shared-path on the x-axis.

5.6 Results

5

Our aim is to evaluate the interaction mode prediction performance of VTP models in an insightful and data-independent way. More specifically, we want to research when mode collapse happens and get insight into the temporal dimension of the predictions. First, we define the parameters used in our experiments in Section 5.6.1. Second, we employ our methodology for finding path-crossing safety-critical interactions on the widely used nuScenes traffic dataset [77], and report interaction statistics in Section 5.6.2. Next, we test four baseline models (described in Section 5.5) and evaluate their performance using our novel evaluation framework in Section 5.6.3. We show that mode collapse happens and provide insights into the temporal evolution of the predictions. Finally, compare our metrics to the traditional distance-based metrics in Section 5.6.4.

5.6.1 Experimental setup

This section discusses the hyperparameters of our evaluation framework[1] to find the safety-critical interactions in a dataset, simulate feasible roll-outs, and test models on our interaction metrics.

First, the PS vectors are computed for all possible agent-pairs, to find the time steps where the agents are on the shared path. We empirically found $d_{\text{collision}} = 1.5\,\text{m}$ to be a reasonable threshold, considering that two narrow cars would be in collision if the distance between their path centerlines is less than 1.5 m. Next, we need to filter out the path-sharing interactions, where there is a big time difference between the agents starting to occupy the same path. After carefully considering various scenarios from the nuScenes dataset, we found $\Delta t_{\text{PS,max}} = 6\,\text{s}$ to be an appropriate threshold.

For simulating the roll-outs, we need to respect acceleration and velocity limits. The absolute longitudinal acceleration limit is set to $|a_{\text{lon}}| \leq 1.47\,\text{m/s}^2$ and the lateral to $|a_{\text{lat}}| \leq 1.18\,\text{m/s}^2$, which is based on the values from [141]. For the accelerations, we also set the maximum velocity equal to the maximum velocity of the scene, thereby implicitly respecting any speed limits or traffic that influences the maximum velocity in the scene.

5.6.2 Interaction statistics nuScenes

We analyzed interactions across the entire train and validation splits of the nuScenes dataset, and applied our methodology to identify safety-critical interactions. In total, we identified 18,299 theoretical interactions across the entire dataset. The theoretical upper limit per scene is calculated as $N(N-1)/2$, considering the symmetry of interactions and the absence of self-pairs. However,

[1] Our code is available online at
https://github.com/MaartenHugenholtz/InteractionEval

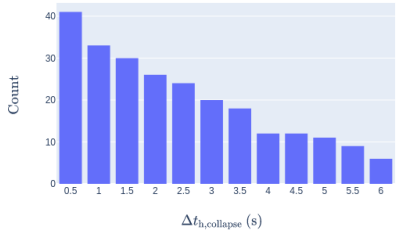


Figure 5.8: Histogram of data samples before the inevitable homotopy state. The samples are prediction frames of safety-critical interaction pairs in the nuScenes validation split.

in reality, only 16,756 theoretical interaction pairs exist, as not all agents are recorded for the full scene duration. After applying our first two interaction criteria, i.e., the agents are not path-sharing at first but are later on, only 730 interaction pairs are left. We characterize the closeness of these interactions in both distance and time in a density heatmap (Figure 5.7). From the figure it becomes clear that the majority of interactions are close, i.e., the time difference is smaller than 6 seconds and the real-time closest distance is smaller than 20m. However, there is also a substantial part of path-sharing interactions, where there is a big time difference between the agents starting to occupy the same path or the distance between them is quite large. Since we are interested in safety-critical interactions, the time difference between the agents should be relatively small. Thus, we apply our third interaction criterion, i.e., $\Delta t_{\text{PS}} \leq 6\,s$, after which only 351 interaction pairs are left in the full train and validation split. That means only 2.1% of the possible interactions are considered safety-critical.

For testing the models on nuScenes, we evaluate them only on the validation split, which contains just 41 safety-critical interaction pairs. After identifying which interactions to evaluate, we determine the evaluation interval $[t_{\text{h,start}}, t_{\text{h,final}}]$ before the homotopy class collapses by employing our methodology for determining the inevitable homotopy state. In Figure 5.8, we present a histogram showing the distribution of samples over their time to the inevitable homotopy state, $\Delta t_{\text{h,collapse}}$. Naturally, this histogram shows a decreasing trend, as the interval during which both agents are recorded in the dataset is relatively short for many interactions. In total, we have just 41 usable interaction pairs, however, for the majority there are just a few samples available before the homotopy class becomes inevitable. There are only 6 pairs for which we can evaluate the predictions a full 6-second prediction horizon before $t_{\text{h,collapse}}$. Next, we will evaluate the models' mode prediction performance on these interaction pairs.

5.6.3 Model intention prediction performance

Predicting the driver's intentions 6 seconds before the interaction is far less important than predicting them 1 second before it happens. On the other hand, correctly predicting the intentions 1 second before the interaction happens, is also a lot easier, as the drivers in the scene have likely already implicitly communicated who takes priority and crosses first, resulting in increased margins and speed differences. To provide insights on the temporal evolution of a model's mode prediction performance, we analyze the mode correct, covered and collapse rates against the time to inevitable homotopy state $\Delta t_{\text{h,collapse}}$, see Figure 5.9. Indeed, we see that, as the interaction comes closer (smaller $\Delta t_{\text{h,collapse}}$), all models are able to more correctly predict the interaction class. That the alternative roll-outs are less likely to happen is also reflected by the higher mode collapse rate of AF for samples closer to the inevitable homotopy state. Although, failing to cover a feasible mode when it is unlikely is not problematic. However, it is worth nothing that in some cases, the models are not even able to

Table 5.2: Interaction mode prediction metrics for AF, CTT, the CV model and the oracle. The rates are evaluated over all interaction-pair samples, whereas the time-based metrics and consistency are computed per interaction-pair sequence and later averaged. We compare the mean time-to-correct/covered mode prediction, as well as the percentage of predictions that are correct from the beginning of the evaluation interval (@$t_{\text{h,start}}$) and the percentage of predictions where the $\Delta T = 0$, meaning the predictions are wrong even just before the inevitable homotopy collapse (@0s). The best metrics in each category are printed **bold** and the second-best *italic*.

Method	T_{pred} (s)	Mode correct rate ↑ (%)	Mode covered rate ↑ (%)	Mode collapse rate ↓ (%)	$\Delta T_{\text{correct}}$ / $\Delta T_{\text{covered}}$			Prediction Consistency ↑ (%)
					mean ↑ (s)	@0s ↓ (%)	@$t_{\text{h,start}}$ ↑ (%)	
AF	6	74.0	*89.3*	*69.8*	1.9 / 1.8	9.8 / 4.9	56.1 / *80.5*	92.7
CV		*80.6*	80.6	100.0	*2.3* / *2.3*	*2.4* / *2.4*	**78.0** / 78.0	**100.0**
Oracle		**86.0**	**100.0**	**18.6**	**2.4** / -	**0.0** / **0.0**	*73.2* / **100.0**	*97.6*
AF	3	83.4	*92.9*	*76.9*	*1.0* / 0.8	12.2 / *4.9*	*70.7* / *87.8*	*95.1*
CTT*		49.3	49.3	100.0	0.1 / 0.1	53.3 / 53.3	40.0 / 40.0	**100.0**
CV		**87.0**	87.0	100.0	0.9 / *0.9*	*7.3* / 7.3	**80.5** / 80.5	**100.0**
Oracle		*86.4*	**100.0**	**13.0**	**1.6** / -	**2.4** / **0.0**	*70.7* / **100.0**	**100.0**

* Note that we were not able to reproduce the numbers reported in CTT's paper, and that some predictions are missing due to the issues discussed in Section 5.5.2.

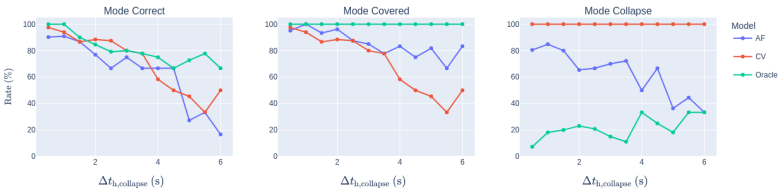


Figure 5.9: Relative mode prediction performance plotted against the time to inevitable homotopy state. From left to right, we consider the correct, covered and collapsed modes. We evaluate AF, the CV model, and the oracle on a prediction horizon of 6 seconds.

correctly predict the interaction class right before the homotopy class becomes inevitable, indicating that mode collapse also occurs in critical situations.

In Table 5.2 the mode correctness, coverage and collapse rates of all models are summarized, as well as the time-based metrics and consistency. First, we will compare the intention prediction performance of AF, the oracle and the CV model on a prediction horizon of 6 seconds, and focus on the time-based metrics. AF correctly predicts the interaction mode at the beginning of the evaluation interval ($@t_{\text{h,start}}$) in 56% of the cases, and otherwise it takes up to 1.9 seconds on average to predict the correct mode. Interestingly, the CV model outperforms the other models and in 78% of the cases it can correctly predict the interaction mode from the beginning by simply extrapolating the vehicle's current trajectory. This shows that in the majority of the cases, the interaction class is a natural evolution of the vehicle's current heading and speed.

Naturally, in the covered category, the multimodal models perform better, as all predictions are considered. AF manages to directly cover the ground truth mode in 80% of the cases, whereas the oracle model achieves a perfect score. The oracle model inherently tries to cover all feasible modes of the interacting agents, but since the models can only predict $K = 5$ futures, it cannot completely mitigate mode collapse in scenes with many agents. The oracle scores a mode collapse rate of 19%, versus the 70% of AF. The CV model inherently suffers from 100% mode collapse due to its unimodal predictions.

Finally, we compare the models, including CTT, on a 3-second prediction horizon, with the results reported in the bottom half of Table 5.2. As explained earlier, CTT's predictions are unimodal and sometimes even missing, resulting in the model's disastrous performance. The mode is predicted correctly right away ($@t_{\text{h,start}}$) for only 40% of the interaction-pairs, and in 53% of the cases, the mode is not predicted at all, resulting in a mean $\Delta T_{\text{correct}}$ of 0.1 seconds. Because of the model's unimodal predictions, it inherently suffers from mode collapse for all scenarios. As we could not reproduce CTT's results, this is not representative of its real performance. However, by testing our methodology on multiple models, we show that it generalizes to other models.

Comparing the other models on the 6-second prediction horizon, we see that the results are slightly different because a shorter prediction horizon changes the number of samples and some predictions may fall in a different homotopy class for the shorter horizon. However, the relative performance differences between the models remain unchanged. Although we cannot compare results from different prediction horizons directly, we demonstrated that our methodology is not limited to a single prediction horizon.

In terms of prediction consistency, all models score high; only in some cases the interaction mode changes inconsistently. The CV model and CTT score 100%, which is more trivial, as they only output a single mode, so inconsistent mode predictions are less likely.

Table 5.3: Distance-based metrics for AF, CTT, the CV model and the oracle, for 3-second and 6-second prediction horizons. The best metrics in each category are printed **bold** and the second-best *italic*.

Method	T_{pred} (s)	ML ADE ↓ (m)	ML FDE ↓ (m)	Joint minADE ↓ (m)	Joint minFDE ↓ (m)
AF	6	3.88	*9.10*	**2.86**	**6.48**
CV model		**3.64**	**9.04**	3.64	9.04
Oracle		*3.84*	9.12	*3.56*	*8.41*
AF	3	1.48	3.00	**1.11**	**2.17**
CTT*		5.93	10.59	5.93	10.59
CV model		**1.22**	**2.68**	*1.22*	2.68
Oracle		*1.45*	*2.85*	1.36	*2.63*

* Note that we were not able to reproduce the numbers reported in CTT's paper, and that some predictions are missing due to the issues discussed in Section 5.5.2.

5.6.4 Distance-based metrics results

In Table 5.3 we compare the models on the traditional distance-based metrics. We report the average and final displacement errors (ADE/FDE) for the most likely (ML) predictions, as well as the joint lower-bound metrics computed for $K = 5$ modes. In contrast to our novel interaction metrics, these are computed over all scenes and time steps of the nuScenes validation split.

Comparing the ML ADE metrics to the ML interaction metrics, we see that the relative performance order remains similar, with the oracle and the CV model performing the best. Interestingly, we see that on the joint metrics, AF performs best, which contradicts with our findings from the interaction metrics. This is partially caused by the fact that the oracle was designed specifically to cover modes of path-crossing vehicles, and not to get the lowest minimum distance errors. More importantly, it also shows that in some cases, the joint distance-based metrics fail to indicate whether the hybrid nature of the predictions is retained by the prediction models.

5.7 Conclusion and Discussion

We introduced a novel evaluation framework to benchmark a model's interaction prediction performance. Our approach explicitly considers the hybrid nature of predictions, with special focus on the discrete interaction modes that capture the high-level behaviors of agent pairs. Our framework simulates alternative interaction modes, and we use this to define a metric for mode collapse on the interaction level. We also use metrics for mode correctness and coverage, and propose time-based variants, that provide insight into the temporal evolution of mode predictions. Uniquely, our method does not evaluate all scenes and frames of a dataset, but only the relevant frames for closely interacting agent-pairs. This reduces the dataset dependency and makes our metrics more insightful and interpretable. We tested four models on the nuScenes dataset [77] and showed that mode collapse happens. Interestingly, a simple constant velocity model outperformed the other models in correctly predicting the interaction mode, showing that in many cases the interaction mode is dictated by the vehicles' current heading and speed. While AgentFormer (AF) manages to produce diverse predictions for each agent, it did not cover all feasible interaction modes between the interacting agents, averaging a mode collapse rate of 70% for the safety-critical interaction pairs. The oracle model, designed to cover all feasible interaction modes, had a mode collapse rate of 20%. Thus,

completely alleviating mode collapse (i.e., covering all feasible interaction modes) is not possible with a finite number of $K = 5$ joint predictions due to the exponentially growing cardinality of the mode space. Although the oracle was superior in covering the interaction modes, it was outperformed by AF on the widely-used joint distance-based metrics, indicating that these metrics do not necessarily capture the model's performance in predicting interaction modes. Finally, we analyzed the temporal evolution of the predictions, and found that both the mode correct and collapse rate increase as the inevitable homotopy state comes closer. In the majority of the scenarios, these collapsed interaction modes do not seem problematic, as they are not likely to happen. However, in a few cases, the models are not able to correctly predict the real interaction class just before the interaction finishes. These incorrectly predicted driver intentions could pose safety concerns for autonomous driving.

While we show that mode collapse occurs, our metrics do not evaluate the severity of the consequences, nor the likelihood of the collapsed modes. In our framework, we simulate feasible futures for interacting agents at every time step, but the model inputs remain the ground truth history of the agents as we replay the scene. Assessing the safety implications of collapsed modes requires a closed-loop simulation setup, in which the predictions are used in a downstream planner. Estimating the likelihood of a collapsed mode could involve comparing the scenario to a distribution learned from traffic data. However, rare but feasible scenarios might be underrepresented and deemed unlikely. Alternatively, planning-like costs could be used to evaluate the safety, comfort, and utility of future roll-outs. Extending our framework to assess the associated risks of false predictions presents an exciting opportunity for future research.

In our framework, we only perform roll-outs and collision checks for pairs of interacting agents. However, in reality, the scenes can be more complex, with multiple agents interacting and influencing each other. While this is a conceptual limitation of our method, the feasibility of our simulations remains valid, as the feasibility is primarily determined by the yielding vehicle's ability to brake before entering the common path, which is not affected by other vehicles.

By applying our methodology to identify safety-critical interactions, we make the evaluation less dependent on the dataset while focusing on the most crucial aspect of driving: the interactions. In the nuScenes dataset, we found that only 2% of the theoretical interactions are considered safety-critical according to our criteria. This finding highlights the need for more interactive datasets and the importance of metrics that are less constrained on the scenarios in a dataset. However, it also reveals a limitation of our approach: we evaluate only real interactions, not hypothetical ones. We opted for this simplistic approach to ensure that the interactions we assess are realistic. Furthermore, simulating all hypothetical interactions would be extremely complex and computationally demanding.

Finally, we analyzed the temporal evolution of the interactions between the critical agent-pairs. For the majority of the pairs, however, there were only a few samples available prior to the interaction, limiting the interpretability of our time-based metrics. This limitation arises because nuScenes is recorded from an on-road viewpoint, constraining the annotations to the range of the ego vehicle. To address this issue, future research could apply our methodology to traffic datasets recorded from a top-down perspective, such as those captured by drones monitoring traffic at intersections [76, 114, 142].

Our novel interaction metrics provide new ways to measure the intention prediction of models in safety-critical interactions. These metrics only take into account the relevant interactions, thereby reducing the dependency on datasets and improving interpretability. Furthermore, our time-based metrics shed light on the temporal evolution of predictions, an aspect that was previously neglected in VTP evaluation. Our new evaluation methodology thus offers new insights and perspectives, helping the holistic evaluation and interpretation of a model's performance. Finally, our evaluation methodology can aid the development of VTP models towards more accurate and consistent hybrid predictions for joint models. Future work should focus on alleviating the aforementioned weaknesses and further generalizing our framework to other datasets and models to establish a benchmark for prediction models.

6

Conclusion and Future Directions

This chapter compiles the conclusions drawn from the work presented in this thesis. It further identifies future directions for the field as well as directions at intersections with related fields of research.

6.1 Conclusion

This thesis set out to address challenges in both the development as well as the evaluation of probabilistic trajectory prediction models. The first two works – ROME (Chapter 2) and TrajFlow (Chapter 3) – of this thesis outperformed comparable state-of-the-art methods at the time. Meanwhile, the last two works on modeling the interactions between agents in joint trajectory prediction models (Chapter 4) and evaluating mode collapse in such models (Chapter 5) addressed gaps in the field which have not been previously studied. Below we discuss each of these works in view of the research questions posed at the beginning of this thesis.

Ensuring good probability density estimation irrespective of distribution type.

Even in the presence of abundant data points from a single distribution, ensuring a good probability density estimation of multi-modal non-Gaussian distributions remains a challenge. Nevertheless, the ROME estimator, introduced in Chapter 2 is a successful method which provides improved density estimations over high-dimensional, multi-modal, and non-Gaussian distributions compared to the state of the art. The proposed method breaks the problem of multi-modal density estimation down to a uni-modal one by leveraging clustering to identify the individual modes. Decorrelation and normalization are then used to distribute the samples with each cluster as close to a Gaussian distribution as possible to facilitate the final density estimation step performed by Kernel Density Estimation using an isotropic Gaussian kernel. The proposed method can thus be applied to varying types of distributions while overcoming issues of over-fitting or over-smoothing which other density estimators exhibit. To achieve even further improvement of the density estimation, particularly for non-Gaussian distributions, it would be worthwhile to investigate the use of more sophisticated density estimators as the internal density estimator of ROME. While relevant to a number of fields, having a robust density estimator is particularly important within the scope of this thesis for evaluating the distribution fit of different prediction models. Regardless of future improvements, ROME is already superior to standard KDE which has been a golden standard in the evaluation of probabilistic trajectory prediction models. As commonly used metrics such as the Negative Log-Likelihood (NLL) rely directly on the underlying density estimation, employing a more robust estimator such as ROME contributes to more reliable and unbiased evaluations of probabilistic trajectory prediction models. To this end, ROME has been integrated into an evaluation framework [116] and applied in evaluations within this thesis. Through this integration, ROME has the potential to serve as a more standardized and fair basis for the evaluation of probabilistic prediction models in future works of other researchers within this field.

Leveraging machine learning to learn explicit distributions over trajectories.

Obtaining accurate probabilistic trajectory predictions is an ongoing topic of research. In light of the end-goal of integrating probabilistic prediction models with probabilistic motion planners, in Chapter 3 we focused on developing a prediction model capable of providing exact likelihoods for the predicted distributions. This model was based on Normalizing Flows, which act as powerful density estimators capable of capturing complex, multi-modal distributions without prior knowledge of the distribution that has to be estimated. NFs have the additional advantage of being able to provide tractable calculation of the exact likelihood for the predicted distributions; something the majority of probabilistic prediction models lack. To improve the density estimation over the trajectories TrajFlow uses a Recurrent Neural Network Auto-Encoder (RNN-AE) to encode the future trajectories and then, rather than learning a distribution directly over the trajectories, learns a distribution over the abstract representations (encodings) of the trajectories. The RNN-AE simplifies the learning procedure by extracting the most relevant features that describe a trajectory. TrajFlow demonstrated

promising results in capturing multi-modal distributions over complete trajectories of humans in traffic compared to state-of-the-art methods of the time.

Within the same time window in which of TrajFlow was introduced, other methods based on NFs also emerged. Although these methods do not capture the distribution over the trajectories directly, they do establish a correlation between the time steps to a similar effect [18, 143]. There is currently no comparison of these models to TrajFlow, which would be valuable for understanding whether these models provide a better distribution fit compared to TrajFlow. Additionally, the practicality of using the trajectory likelihoods provided by NF models in motion planning has yet to be studied. This is an important step that is needed for understanding the viability of these kinds of prediction models being applied in autonomous vehicles.

Modeling the interaction between agents in joint prediction models.

While there are a number of joint trajectory prediction models which strive to capture the future evolution of the complete scene, it remains unclear from the state-of-the-art what the best way to model the interactions between the agents is. In Chapter 4 we investigated a number of ways to represent the interactions between agents. Commonly used representations for the interaction between agents are typically based on heuristics such as Euclidean distance or predicting the chance of the trajectories of two agents crossing in the future. We leveraged these representations to construct interaction graphs which we then used to factorize the joint distributions. Using this factorization, we expanded the concept of learning distributions over trajectories as proposed in Chapter 3 to learning the joint distribution of all agents in a scene. In this manner, we could study the impact of the different approaches for representing the influence between agents on the final learned distribution. Our findings showed that more often than not, simply allowing a network to establish interactive connections between agents based on data has a detrimental effect on performance. Instead, having well defined interactions can often bring about a clear boost in performance. This all points toward the need for better understanding how to represent the influence between agents as well as exploring how to leverage knowledge from other fields such as neuroscience and human behavior modeling to this end. A deeper discussion on this matter is provided in the following section on future direction.

Evaluating the predicted modes for multi-modal joint distributions.

Finally, if we are already developing joint prediction models with the intention of capturing the likelihood of potential future interactions, it is important to also be able to evaluate if these different interactions are actually being predicted. While mode collapse is a known problem in the field of trajectory prediction, there have thus far been no evaluation methods to systematically detect and measure this. In Chapter 5 we proposed a novel evaluation framework based on homotopy classes that assesses the modes in joint trajectory predictions, focusing on safety-critical interactions. We introduced metrics for mode collapse, mode correctness, and coverage, emphasizing the sequential dimension of predictions. When looking at the sequential dimension, although prediction accuracy improves closer to interaction events, there are still cases where the models are unable to predict the correct interaction mode, even just before the interaction mode becomes inevitable. Our framework further allowed us to identify models which suffer from inconsistent predictions, i.e. changing the most likely mode more than once over time for interacting agent pairs. In both cases, if such shortcomings are not treated either through improved predictions or fail safes on the motion planning side they can cause undesired behavior of the AV. Currently, the proposed framework has only been applied to vehicle-vehicle interactions. In order to make it more widely applicable for evaluating trajectory prediction models, it would be necessary to expand it to other agent types. Additionally, the framework only looks at situations where interactions between agents actually happened. It does not consider situations in which an interaction could have occurred. For a more complete evaluation, it would be beneficial to expand the framework to account for those situations as well.

Nevertheless, our proposed framework is a first step towards tools to help developers of AVs identify the shortcomings of their models in terms of the quality of the predicted distribution.

6.2 Future Directions

The topic of human trajectory prediction is by far not a solved problem yet. Connecting to the contributions made in this thesis as well as adjacently related fields, we identify three main research directions which would be valuable to explore. The directions pertain to better understanding human interaction in traffic, establishing more holistic evaluation approaches, and lastly how developments in infrastructure could support intelligent vehicles. We explore the three directions in more detail in the following sections.

6.2.1 Understanding Interactions Between Traffic Participants

A general issue with data used for training prediction models is that the data is purely observational. As a result, the majority of prediction models will base their predictions on correlations between input features and given outputs. While this is a valid approach in theory, due to the limited amount of data and computational resources it is not the most reliable approach for ensuring good quality predictions. This was also observed in Chapter 4, where already using simple heuristics to provide structure to the joint distribution problem resulted in a performance boost as opposed to relying solely on the network to extract the relevant information from the data. An important and highly dynamic component of the input and output data are the features of the agents in the scene. These agents are continuously evaluating their surroundings and adjusting their behavior accordingly, affecting their surroundings in turn.

In order to reduce the necessary amount of data needed to understand how such interactions unfold, it would be beneficial to first identify the underlying fundamental mechanics governing people's reasoning in traffic. While there is already extensive literature in the field of human behavior modeling striving to address this matter, models developed in this particular field are generally tailored to specific scenarios (e.g. intersections, roundabouts, pedestrian crossings, etc.) [144, 145], and typically simulate a single outcome for a given parametrization of the model. Additionally, a number of models focusing on the interaction between agents are often limited to pairwise interactions [146–148]. Even more generalized approaches such as the CEI model [118], which identified a reasoning loop comprising of belief-act-communicate, or the risk-based driver model [117] which identified that modeling behavior in accordance to risk fields gives rise to human-like emergent behavior are limited in their scalability to being applied in dense traffic scenes. Additionally, these models still make assumptions on the model parameters, which is valid for generating control inputs for human-like behavior but is not sufficient for complete and accurate predictions of another human's behavior.

As such, interesting directions of inquiry would include:

- developing network structures informed by human behavior models for improved scalability,
- identifying the parameters of human behavior models in real-time from observations,
- and on the topic of communication, further identify which modes of communication are used, and how much weight agents place on these modalities of information in different situations.

6.2.2 Evaluation Beyond Prediction

Establishing metrics for trajectory prediction models continues to be an ongoing direction for research, since existing metrics provide an incomplete indication of predictive performance. The shortcomings of standard metrics in the field of trajectory prediction are best observed upon integration with

motion planners. Upon integration, prediction models with the best metric performance may not necessarily result in the best driving behavior. [149]

With this in mind, we should also be striving towards system level evaluation benchmarks which evaluate the system at different integration levels such as the integration of prediction and planning. Having an established benchmark within the research community would enable developers of upstream modules, such as prediction modules, to evaluate how their modules interact with downstream modules such as the planning module. The goal of the prediction modules in an AV stack is to better inform the planner, making the interaction of the two modules more important than the performance of the prediction module in isolation. Additionally, when testing the prediction module in isolation, as is common to do in the field, one cannot account for the effect of the planner on the predictions nor on potential changes in the surrounding agents' behaviors. This is because the training data gathered generally contains data of human participants and not AV-related data such as the planned and executed maneuvers. Since gathering data for every new motion planner, however, would be both costly and potentially dangerous, having an integration level benchmark would enable a more holistic evaluation of prediction and planning.

Developing such a benchmark brings about its own set of challenges. For one, different types of metrics or tools would be required to disentangle the contribution of the different modules to the final behavior of the vehicle. Secondly, to reduce mismatch between performance in the benchmarking environment and performance upon real-world deployment it is important to consider how other agents will react to the AV and how this may in turn affect the AV. While there do exist works towards benchmarking frameworks for motion planners with consideration towards reactive agents, such as NuPlan [150] and Waymax [151], the surrounding agents are still modeled using the intelligent driver model (IDM). Even though IDM captures human behavior reasonably well, it is considered to be a bit of an idealistic representation of human behavior [152]. This raises the further question of whether IDM should continue to be used for closed-loop evaluation, or if other models – that may have yet to be developed – would be more adequate. Lastly, to develop such a benchmark in a way that can benefit researchers globally, more focused work on modeling the cultural differences in traffic participant behavior would have to be undertaken.

6.2.3 Developing Interfaces and Infrastructures That Support Intelligent Vehicles

Research being done in the fields of perception and prediction does not need to be limited to the development of a fully autonomous system. These systems can, for example, be used to aid drivers in the form of artificial co-pilots. Often when a person rides in the front passenger seat, this person is also an active traffic participant, monitoring the surroundings and providing input in situations of limited visibility or busy traffic. Even "simple" prediction models such as gap acceptance models could be integrated in the form of driver assistance to aid with highway merging or unprotected left turns in urban settings. Before integration of any such models, research into the best manner of interfacing the information from these models would have to be conducted to ensure efficient communication and minimal distraction of the driver. A second pair of eyes, or a system in their stead, that evaluates the situation around a driver and provides input when needed already has the potential to reduce accidents caused by drivers misjudging their surroundings.

Similarly, one of the goals of AVs is to improve road safety beyond the level of average human drivers. One particularly tricky aspect of urban driving are frequent occlusions of traffic participants – especially vulnerable road users – either by other traffic participants or simply due to the road infrastructure. While a solution could be to build models which reason about what might be behind these occluded regions, another worthwhile line of inquiry is looking into how to expand existing infrastructures or vehicles to communicate information about these occluded regions. Having a system for communication could in a first step be used to aid human decision making in highly

occluded scenarios and in a further step give more sophisticated modules, such as prediction and planning modules in an AV a more complete picture of its environment as well. This line of inquiry brings with it a number of matters which would need to be considered, from limited bandwidth, communication delays, as well as cybersecurity before it could be introduced in the real world.

6.3 Closing Remarks

Whichever direction one takes, the topic of predicting human behavior and in turn making decisions in traffic remains a complex one. The dynamic and interactive nature of the behavior of traffic participants introduces significant challenges in both developing and evaluating prediction models.

The findings and methods of this thesis contribute to the overarching goal of making road transport safer. While the prediction models proposed here are not yet ready for direct deployment in real-world AVs, they provide important findings relevant to improving probabilistic predictions. From understanding how to better represent the output distributions of individual agents, to modeling their joint futures. These advances bring us closer to systems capable of reasoning about human intent and uncertainty in a way that is more in line with human reasoning, which in turn can help improve both the transparency of the decision process as well as the data efficiency of the training. This is particularly true if we pursue the development of prediction models grounded in human behavior research. These models are inherently more transparent than purely data driven approaches due to the structure they introduce. This added transparency can make it easier to comprehend the decision process underlying the final prediction. Doing so can not only contribute to improved trust in the system, but can also help in auditing the system both prior to deployment as well as in the case of failure upon deployment.

In terms of evaluating prediction models, the methods proposed in this thesis contribute to the improved evaluation of the models in terms of their predicted distributions. The integration of ROME into a more general evaluation framework [116] has the potential to reach a wider audience of prediction model developers. Through the introduction of a more sophisticated density estimator in the stead of standard KDE, developers stand to gain a fairer and more accurate comparison of their models to state-of-the-art models. This in turn also ensures that the performance on probabilistic metrics corresponds to the actual quality of the predicted distributions. Additionally, our proposed framework for the evaluation of safety-critical interactions provides a further tool to developers of prediction models. By focusing on safety-critical interactions, we zoom in on one the situations where accurate predictions can be crucial to the safe and comfortable navigation of an AV. Not only are safety-critical interactions under-represented in a majority of real-world datasets, the performance of models on these interactions are generally averaged out by the performance on the complete dataset. Meanwhile, safety-critical interaction scenarios are precisely the scenarios in which drivers stand to benefit from autonomous systems, as these are the scenarios with a greater potential for accidents [153, 154]. While the proposed framework has its limitations, it provides a step towards a more focused evaluation of prediction models, and hopefully some food for thought on directing research efforts towards scenarios where prediction quality more directly contributes to road safety.

Furthermore, the interplay between prediction, planning, and communication – whether among AVs or between vehicles and human drivers – highlights the importance of holistic system design. Moving forward it is important to create tools, standards and infrastructures for developing and testing components of the system, thus facilitating a holistic system design even for individual, independent researchers. It is through such means that we can collectively contribute to this highly interdisciplinary field, which spans not only engineering and behavioral sciences but also aspects relating to ethics and legislature. Such interdisciplinary collaborations are relevant to ensuring not only improved safety on the roads but also to consider cultural differences for more seamless and broader acceptance of AVs.

A

Appendix A: Inner Workings of ROME and Further Evaluations

A.1 Clustering within the OPTICS Algorithm

In our implementation of the OPTICS algorithm [43], we follow the standard approach in regard to the generation of the reachability distances $\boldsymbol{R}_N$ and the corresponding ordered samples $\boldsymbol{X}_{I,N}$. However, we adjusted the extraction of the final set of clusters $\boldsymbol{C}$. Namely, in the standard OPTICS algorithm, one would have predetermined either the use of **DBSCAN** or **ξ-clustering** together with the corresponding parameters (respectively ϵ and ξ).

DBSCAN assumes that a cluster is defined by the fact that all of its samples are closer together than a predefined distance, which is the same for all clusters in a dataset. However, this assumes that clusters generally have a similar density, which might not always be the case. Therefore, ξ-clustering is designed to select clusters, whose member samples are significantly closer together compared to the surrounding samples, which means that this algorithm can recognize clusters with varying densities. However, it is more susceptible to noise.

Consequently, it is likely impossible to select one certain method and parameter that gives optimal performances for all potential cluster configurations. However, as this extraction is far cheaper than the previous ordering of samples, we run a large number of different cluster extractions, and use the Silhouette score to determine the optimal set of clusters. This approach makes the clustering of the samples more robust.

A.2 Likelihood factors

Given two average log likelihoods $\widehat{L}_A$ and $\widehat{L}_B$ of two density estimation methods A and B, we can calculate a so-called *factor* $\mathcal{F}$ that expresses how much more likely method A is at reproducing the underlying distribution p compared to method B:

$$\mathcal{F} = \frac{1}{100}\sum_{i=1}^{100}\mathcal{F}_i = \frac{1}{100}\sum_{i=1}^{100}\exp\left(\widehat{L}_{A,i} - \widehat{L}_{B,i}\right),$$

A

for 100 repeated evaluations i. We then measure the statistic significance of $\mathcal{F} > 1$ using a standard t-test based on the 100 $\mathcal{F}_i$ values.

A.3 Clustering Performance

An important component of ROME is the OPTICS algorithm, which we utilize for identifying the clusters (i.e., the modes) within the data. An important aspect of OPTICS is a sensible choice of k_c, a variable used to guard against randomness in the estimation of reachability distances (see equation (2) in the main text). In this work, we used the following rule of thumb to determine these parameters, where N is the number of samples and M the number of dimensions:

$$k_c = \min\left\{k_{\max}, \max\left\{k_{\min}, \frac{NM}{\alpha_k}\right\}\right\}.$$

This part contains three hyperparameters – namely $k_{\min}$, $k_{\max}$, and α_k. Firstly, $k_{\min}$ is needed to ensure that the method is stable, as a too low k_c would make the subsequent clustering vulnerable to random sampling fluctuations. Meanwhile, $k_{\max}$ ensures that the reachability distances are actually based on only local information, and are not including points from other modes. Lastly, the term NM/α_k is used to ensure an independence from the number of samples, while allowing for the higher number of samples needed in higher-dimensional spaces.

The hyperparameters were then empirically selected such that good clustering could be guaranteed over multiple datasets with a different number of samples, resulting in $k_{\min} = 5$, $k_{\max} = 20$, and $\alpha_k = 400$. With those values, we obtain satisfactory clustering performance across all datasets even when the number of samples is greatly reduced (Figures A.1 and A.2). The only exception to this is the Two Moons dataset with 100 samples. However, given the gaps in the intended clusters, this could be expected.

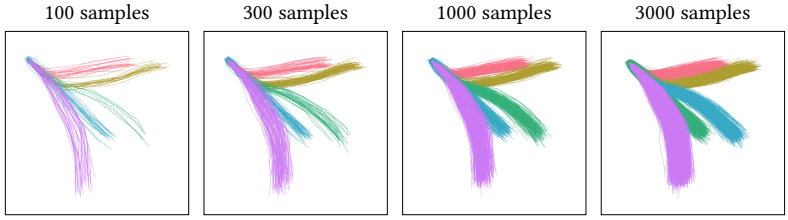


Figure A.1: Clustering results of OPTICS on the Trajectories dataset when using the silhouette score for the selection criterion

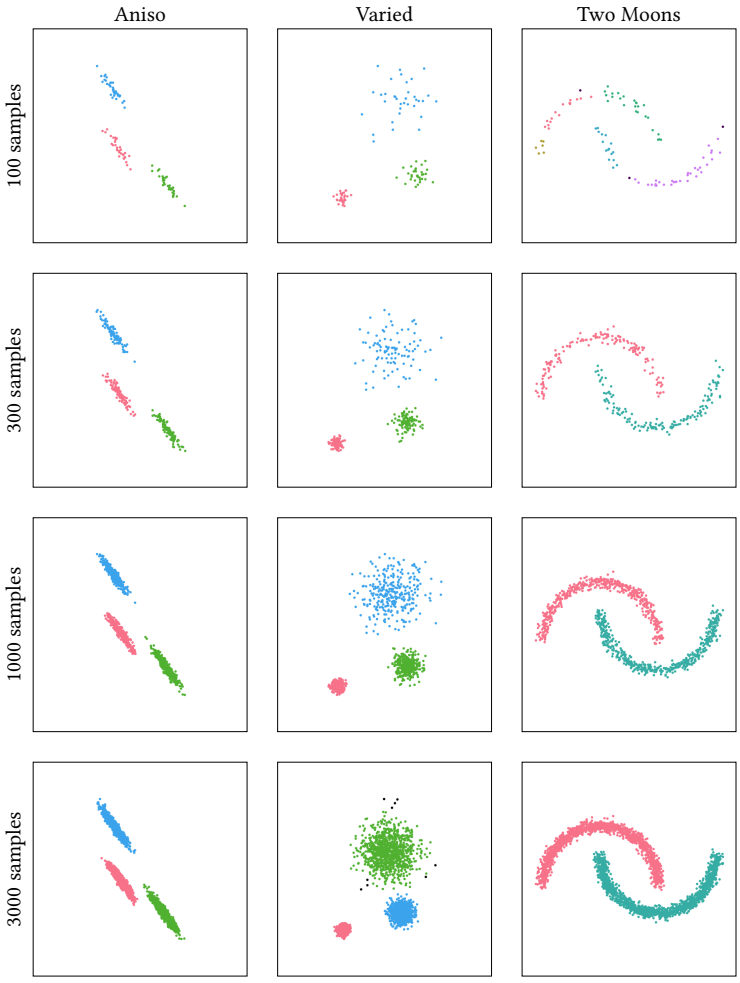


Figure A.2: Clustering results of OPTICS on 2D distribution datasets when using the silhouette score for the selection criterion.

A

Table A.1: Baseline Comparison ($D_{\mathrm{JS}_{\mathrm{true}}} \downarrow_0^1$; ground truth distributions) - marked in red is a case of notably poor performance; best values are underlined

Distrib.	ROME	MPW	VC
Aniso	$\underline{0.007}_{\pm 0.001}$	$0.016_{\pm 0.001}$	$0.143_{\pm 0.003}$
Varied	$\underline{0.078}_{\pm 0.003}$	$0.084_{\pm 0.003}$	$0.087_{\pm 0.003}$

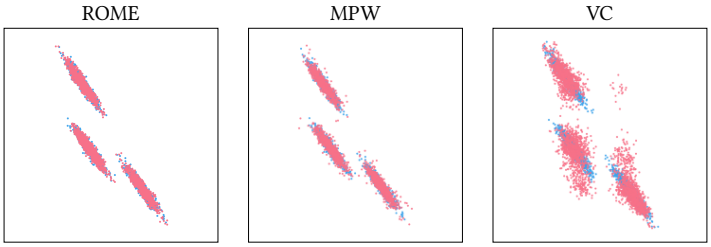


Figure A.3: Samples obtained with ROME, MPW and VC (pink) contrasted with samples from p (blue); Aniso.

A.4 Comparison to Ground Truth Distributions

In order to get a better idea of the goodness of fit of the different density estimators, we also compare the distributions $\widehat{p}_1 = f(\boldsymbol{X}_1)$ obtained from the estimator f to the ground truth distribution p using the Jensen-Shannon divergence $D_{\mathrm{JS}}(\widehat{p}_1 \| p)$. To simplify notation, this metric will be referred to as $D_{\mathrm{JS}_{\mathrm{true}}}$. The ground truth distribution p is only available for Aniso and Varied.

In the baseline comparison we observe that our estimator ROME achieves the lowest $D_{\mathrm{JS}_{\mathrm{true}}}$ values, particularly in the case of the Aniso distribution (Table A.1). We further see that f_{VC} is unable to fit the Aniso dataset well, which is supported by further visual inspection (Figure A.3).

Within our ablation study on the Aniso dataset we clearly observe the importance of both decorrelation and normalisation in the case of highly correlated features (Table A.2). Meanwhile, in our ablation study on the Varied dataset, we continue to observe the importance of clustering for a good fit on data with varying densities across modes, as indicated by the high $D_{\mathrm{JS}_{\mathrm{true}}}$ values when no clustering is used (Table A.3). In the ablation cases, as we cannot guarantee that using f_{kNN} results in a normalized distribution, the corresponding values should be considered with caution.

Table A.2: Ablations ($D_{\mathrm{JS}_{\mathrm{true}}} \downarrow_0^1$, Aniso; comparison to the ground truth distribution) - When clustering, decorrelation and normalization improve results for distributions with high intra-mode correlation. ROME highlighted in gray.

	Norm.				No norm.		
	Decorr.		No decorr.				f_{GMM}
Cluster.	f_{KDE}	f_{kNN}	f_{KDE}	f_{kNN}	f_{KDE}	f_{kNN}	
Silhouette	0.007±0.001	0.014±0.001	0.039±0.001	0.020±0.001	0.054±0.001	0.020±0.001	0.001±0.000
DBCV	0.009±0.002	0.014±0.001	0.037±0.001	0.022±0.001	0.061±0.003	0.023±0.002	0.008±0.003
No clus.	0.095±0.001	0.015±0.001	0.245±0.002	0.022±0.001	0.039±0.001	0.023±0.001	0.530±0.002

Table A.3: Ablations ($D_{\mathrm{JS}_{\mathrm{true}}} \downarrow_0^1$, Varied; comparison to the ground truth distribution) - Clustering is essential to improving results for distributions with varying mode densities. ROME highlighted in gray.

	Norm.				No norm.		
	Decorr.		No decorr.				f_{GMM}
Cluster.	f_{KDE}	f_{kNN}	f_{KDE}	f_{kNN}	f_{KDE}	f_{kNN}	
Silhouette	0.078±0.003	0.083±0.003	0.078±0.003	0.083±0.003	0.070±0.003	0.083±0.003	0.074±0.003
DBCV	0.074±0.003	0.084±0.003	0.074±0.003	0.084±0.003	0.065±0.003	0.085±0.003	0.068±0.003
No clus.	0.173±0.004	0.081±0.003	0.183±0.004	0.080±0.003	0.074±0.003	0.080±0.003	0.540±0.003

A

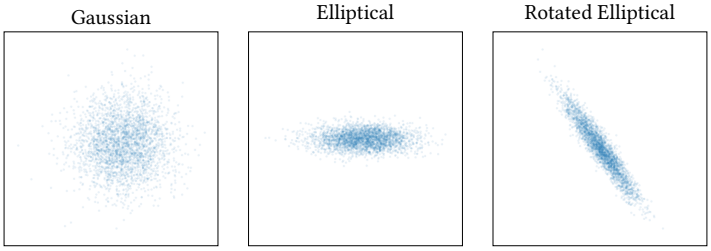


Figure A.4: Samples from the two-dimensional uni-modal distributions.

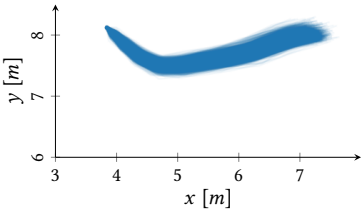


Figure A.5: Samples from the uni-modal pedestrian trajectory distribution; obtained by selecting a single mode from the original multi-modal distribution.

A.5 Comparison on Uni-modal Distribution

In order to verify that ROME is not tailored only to multi-modal distributions, but can in fact be applied to simpler uni-modal distributions as well, we conduct experiments on the Gaussian, Elliptical and Rotated Elliptical distributions (Figure A.4), as well as on a more complex multivariate distribution using a single mode of our Trajectories dataset (Figure A.5). We perform both the baseline comparison as well as the ablation study as done for the multi-modal datasets presented in the main body of the paper.

Within the baseline comparison, ROME continues to achieve high performance compared to the baselines across all three metrics on the uni-modal datasets (Table A.4) and are further supported by the results of $D_{\mathrm{JS}_{\mathrm{true}}}$ on the ground truth distributions of the two-dimensional uni-modal distributions (Table A.5). The only exception is the Wasserstein-based metric $\widehat{W}$, where VC achieves slightly better performance. However, when this information is coupled with the values of D_{JS} and $\widehat{L}$, we can conclude that VC struggles in fitting distributions with strongly correlated features as seen by the results on the Rotated Elliptical distribution and the multivariate Trajectories distribution.

Within the ablation study, it is evident that clustering has little to no effect on the final density estimation, which is to be expected since there is only a single cluster in these datasets. Unsurprisingly, in the case of the two-dimensional uni-modal distributions, using f_{GMM} achieves the best results since the underlying distributions are Gaussian by nature. However, ROME still leads to a good fit on these simple distributions. The ablations on the Elliptical and Rotated Elliptical distributions once more demonstrate the importance of normalizing as well as decorrelating the data prior to the density estimation, particularly when using f_{KDE} as the underlying density estimator (Tables A.6 and A.7).

Table A.4: Baseline Comparison, uni-modal – marked in red are cases with notably poor performance; best values are underlined.

	$D_{\text{JS}} \downarrow_0^1$			$\widehat{W} \to 0$			$\widehat{L} \uparrow$		
	f_{ROME}	f_{MPW}	f_{VC}	f_{ROME}	f_{MPW}	f_{VC}	f_{ROME}	f_{MPW}	f_{VC}
Gaussian	0.006±0.001	0.023±0.002	0.007±0.001	−0.02±0.09	−0.26±0.05	−0.01±0.11	−2.84±0.02	−2.87±0.02	−2.85±0.02
Elliptical	0.005±0.001	0.023±0.002	0.007±0.001	0.05±0.17	−0.29±0.08	0.04±0.16	−1.64±0.02	−1.66±0.02	−1.64±0.02
Rot. Ellip.	0.005±0.001	0.023±0.002	0.003±0.001	0.04±0.20	−0.35±0.11	0.84±0.32	−1.42±0.02	−1.45±0.02	−1.55±0.01
Uni. Traj.	0.001±0.000	0.003±0.000	0.801±0.003	1.69±0.01	2.33±0.01	1.60±0.11	32.46±0.02	28.20±0.02	−193.88±17.5

Table A.5: Baseline Comparison ($D_{\text{JS}_{\text{true}}} \downarrow_0^1$; ground truth uni-modal distributions) - best values are underlined.

Distrib.	ROME	MPW	VC
Gaussian	0.004±0.001	0.012±0.001	0.004±0.001
Elliptical	0.004±0.001	0.012±0.001	0.004±0.001
Rot. Elliptical	0.004±0.001	0.013±0.001	0.029±0.001

A

Table A.6: Ablations ($D_{\mathrm{JS}_{\mathrm{true}}} \downarrow_0^1$, Elliptical) - Normalization improves the fit for anisotropic Gaussian distributions. ROME highlighted in gray.

Clustering	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.	f_{GMM}
Silhouette	0.004±0.001	0.004±0.001	0.017±0.001	0.000±0.000
DBCV	0.004±0.001	0.004±0.001	0.017±0.001	0.000±0.000
No clusters	0.004±0.001	0.004±0.001	0.017±0.001	0.000±0.000

Table A.7: Ablations ($D_{\mathrm{JS}_{\mathrm{true}}} \downarrow_0^1$, Rotated Elliptical) - Normalization and decorrelation improves the fit for distributions with strongly correlated features. ROME highlighted in gray.

Clustering	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.	f_{GMM}
Silhouette	0.004±0.001	0.027±0.001	0.038±0.001	0.000±0.000
DBCV	0.004±0.001	0.027±0.001	0.038±0.001	0.000±0.000
No clusters	0.004±0.001	0.027±0.001	0.038±0.001	0.000±0.000

Finally, the results on the uni-modal Trajectories dataset exhibit similar trends to the ones on the four multi-modal datasets presented in the main body of the paper. These trends include a tendency towards over-smoothing in the case of using f_{GMM} as the underlying density estimator in the case of non-normal distributions (Table A.8). Additionally, we observe that normalizing and decorrelating the data prior to performing the density estimation plays an important role when using f_{KDE} as the underlying density estimator – which was selected for ROME (Tables A.8-A.10).

Table A.8: Ablations ($\widehat{W} \to 0$, Uni-modal Trajectories) - Excluding normalization or using f_{GMM} as the downstream estimator is not robust against non-normal distributions. ROME highlighted in gray.

Clustering	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.	f_{GMM}
Silhouette	1.69±0.01	1.68±0.02	22.39±0.10	2.20±0.03
DBCV	1.69±0.01	1.68±0.02	23.18±2.38	2.20±0.03
No clusters	1.69±0.01	1.68±0.02	22.39±0.09	2.20±0.03

Table A.9: Ablations ($\widehat{L} \uparrow$, Uni-modal Trajectories) - Excluding normalization is not robust against non-normal distributions. ROME highlighted in gray.

Clustering	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.	f_{GMM}
Silhouette	32.46±0.02	31.87±0.10	−13.85±0.00	28.61±0.02
DBCV	32.46±0.02	31.87±0.10	−14.51±1.96	28.61±0.02
No clusters	32.46±0.02	31.87±0.10	−13.85±0.00	28.61±0.02

Table A.10: Ablations ($D_{\text{JS}} \downarrow_0^1$, Uni-modal Trajectories) - Using decorrelation is necessary to avoid overfitting when using normalization.

Clustering	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.	f_{GMM}
Silhouette	0.013±0.004	0.166±0.063	0.000±0.000	0.005±0.003
DBCV	0.013±0.004	0.166±0.063	0.005±0.031	0.005±0.003
No clusters	0.013±0.004	0.166±0.063	0.000±0.000	0.005±0.003

B

Appendix B: The Encoding of Past Behavior in TrajFlow

B.1 Past Trajectories

The encoder of the past behavior ϕ_{RNN} encodes the past trajectory x_{tar} of the single target agent whose future is to be predicted. This function was taken from the implementation of [73] and is identical to the encoder E_{RNN} (see Sec. 3.3.2), except that instead of $d = M = m = 20$, we used $d = M = m = 64$ for the TrajFlow variants and $d = M = m = 16$ in FloMo, as per the original implementation.

B.2 Static Environment

For encoding a gray-scale image of the static environment E, which has been rotated to align with the target agent's heading, we used a CNN function ϕ_{CNN}. For this, $L_{\text{CNN}} = 3$ convolutional layers $\phi^{(l)}_{\text{CNN}}$ are used within this network with a kernel of size 5 and a stride of 4. The first two layers additionally have a zero-padding of size 1 around the image. With this, an initial input of size $h^{(0)} \times w^{(0)} = 156 \times 257$ and $c^{(0)} = 1$ channel is transformed first into a representation with $c^{(1)} = 8$ and $h^{(1)} \times w^{(1)} = 39 \times 64$, then into a representation with $c^{(2)} = 16$ and $h^{(2)} \times w^{(2)} = 10 \times 16$ and lastly into an output representation with $c^{(3)} = 32$ and $h^{(3)} \times w^{(3)} = 2 \times 3$. This output $\phi^{(3)}_{\text{CNN}}$ is then flattened and passed through a two-layer dense network. The first linear layer transforms the input into a hidden state of length 128, while the second linear layer produces the final encoding of the image of size $M_{\text{CNN}} = 64$.

B.3 Social Interactions

To encode interactions, we use a GNN function ψ_{GNN} that processes all past trajectories $\mathbf{x} = \{\mathbf{x}_{\text{tar}}, \mathbf{x}_1, \dots \mathbf{x}_{n-1}\}$. There, in the first step, the past trajectory $\mathbf{x}_a$ of each agent a of the n agents is encoded using a GRU-based function $\psi_{\text{RNN},c}$ (structure is identical to ϕ_{RNN}; Sec. B.1). This network is shared between all agents of each class $c \in C = \{\text{veh.}, \text{ped.}, \dots\}$, i.e. there is for instance one network $\psi_{\text{RNN, veh.}}$ to encode the past of vehicles. A linear embedding layer $\psi_{\text{em}} : \mathbb{R}^m \to \mathbb{R}^{M_{\text{GNN}}}$ is then applied to each encoded past trajectory, with $\widetilde{\mathbf{x}}^{(0)}_a = \psi_{\text{em}}\left(\psi_{\text{RNN},c_a}(\mathbf{x}_a)\right)$.

In the GNN, each of the n agents is seen as a node, with n^2 unidirectional edges being established between all nodes. Based on this, L_{GNN} layers $\psi^{(l)}_{\text{GNN}}$ are applied to this network to update the node

states $\widetilde{\mathbf{x}}^{(l)} = \left\{\widetilde{\mathbf{x}}_{\text{tar}}^{(l)}, \widetilde{\mathbf{x}}_{1}^{(l)}, \ldots, \widetilde{\mathbf{x}}_{n-1}^{(l)}\right\}$:

$$\widetilde{\mathbf{x}}^{(l)} = \psi_{\text{GNN}}^{(l)}\left(\widetilde{\mathbf{x}}^{(l-1)}\right)$$

The update starts with calculating the message $\mathbf{m}_{b,a}^{(l)}$ from agent b to agent a for every possible connection, using the message network $\psi_{\text{M}} : \mathbb{R}^{2M_{\text{GNN}}+2|C|+1} \to \mathbb{R}^{M_{\text{GNN}}}$:

$$\mathbf{m}_{b,a}^{(l)} = \psi_{\text{M}}\left(\widetilde{\mathbf{x}}_{b}^{(l-1)} \oplus \widetilde{\mathbf{x}}_{a}^{(l-1)} \oplus C_b \oplus C_a \oplus \|\mathbf{x}_a - \mathbf{x}_b\|\right),$$

where the last three terms are the graph's edge features between agents a and b, with $C_a, C_b \in \mathbb{R}^{|C|}$ being the one-hot encoding of class c_a and c_b respectively. Those incoming messages are then aggregated at each node:

$$\mathbf{m}_{a}^{(l)} = \sum_{b} \mathbf{m}_{b,a}^{(l)}$$

Lastly, the state of each node is updated, using an update network $\psi_{\text{U}} : \mathbb{R}^{2M_{\text{GNN}}} \to \mathbb{R}^{M_{\text{GNN}}}$

$$\widetilde{\mathbf{x}}_{a}^{(l)} = \psi_{\text{U}}^{(l)}\left(\widetilde{\mathbf{x}}_{a}^{(l-1)} \oplus \mathbf{m}_{a}^{(l)}\right) + \widetilde{\mathbf{x}}_{a}^{(l-1)}$$

After being propagated through all L_{GNN} layers $\psi_{\text{GNN}}^{(l)}$, the final output of ψ_{GNN} is

$$\frac{1}{n}\sum_{i} \widetilde{\mathbf{x}}_{a}^{(L_{\text{GNN}})}$$

For our work, we chose to set $L_{\text{GNN}} = 4$ and $M_{\text{GNN}} = 32$.

References

References

[1] Pete Thomas, Andrew Morris, Rachel Talbot, and Helen Fagerlind. Identifying the causes of road crashes in europe. *Annals of advances in automotive medicine*, 57:13, 2013.

[2] Road safety statistics in the eu https://ec.europa.eu/eurostat/statistics-explained/index.php?title=road_safety_statistics_in_the_eu. Accessed: 2025-04-03.

[3] Rins de Zwart, Reinier J Jansen, Cheryl Bolstad, Mica R Endsley, Petya Ventsislavova, Joost de Winter, and Mark S Young. When is more actually better? expert opinions on assessment of situation awareness in relation to safe driving. *Transportation Research Part F: Traffic Psychology and Behaviour*, 108:54–72, 2025.

[4] Paweł Droździel, Sławomir Tarkowski, Iwona Rybicka, and Rafał Wrona. Drivers' reaction time research in the conditions in the real traffic. *Open Engineering*, 10(1):35–47, 2020.

[5] Cristina Bustos, Albert Sole-Ribalta, Neska Elhaouij, Javier Borge-Holthoefer, Agata Lapedriza, and Rosalind Picard. Analyzing the visual road scene for driver stress estimation. *IEEE Transactions on Affective Computing*, 2025.

[6] Disability in the eu: facts and figures https://www.consilium.europa.eu/en/infographics/disability-eu-facts-figures/#0. Accessed: 2025-04-03.

[7] Disabled road users [what europe does for you] https://epthinktank.eu/2018/08/06/disabled-road-users-what-europe-does-for-you/. Accessed: 2025-04-03.

[8] https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Population_structure_and_ageing. Accessed: 2025-04-03.

[9] Mobility poverty requires inclusive policies https://www.tno.nl/en/newsroom/insights/2022/10/mobility-poverty-inclusive-policies/. Accessed: 2025-10-22.

[10] How to guarantee public transport inclusiveness considering aging, gender, disabilities and reduced mobility https://transport.ec.europa.eu/document/download/d19bd3a5-d5c8-4de9-a248-a035078f223f_en?filename=egum%20recommendations_pt%20subgroup_topic%204a.pdf. Accessed: 2025-10-22.

[11] The u.s. fsd investigation and its repercussions for europe: Autonomy liability and regulation https://www.teslaacessories.com/blogs/news/the-u.s.-fsd-investigation-its-repercussions-for-europe-autonomy-liability-and-regulation. Accessed: 2025-10-22.

[12] Self-driving taxis are speeding ahead in america—so why is europe still waiting? https://tech.yahoo.com/transportation/articles/self-driving-taxis-speeding-ahead-060000376.html. Accessed: 2025-10-22.

[13] RasaBasa. Old ape driving down a narrow street in the center of the old town of tropea. vintage car for transportation and tourist attraction. travel vacation in calabria, southern italy. [photograph] https://stock.adobe.com/nl/images/old-ape-driving-down-a-narrow-street-in-the-center-of-the-old-town-of-tropea-vintage-car-for-transportation-and-tourist-attraction-travel-vacation-in-calabria-southern-italy/535592889. Accessed: 2025-10-24.

[14] James131/Wirestock Creators. Swindon's iconic magic roundabout. [video still] https://stock.adobe.com/nl/video/swindon-s-iconic-magic-roundabout/650889434. Accessed: 2025-10-24.

[15] Giuseppe M Ferro and Didier Sornette. Stochastic representation decision theory: How probabilities and values are entangled dual characteristics in cognitive processes. *Plos one*, 15(12):e0243661, 2020.

[16] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

[17] Renhao Huang, Hao Xue, Maurice Pagnucco, Flora D Salim, and Yang Song. Vision-based multi-future trajectory prediction: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[18] Takahiro Maeda and Norimichi Ukita. Fast inference and update of probabilistic density estimation on trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9795–9805, 2023.

[19] Abduallah Mohamed, Deyao Zhu, Warren Vu, Mohamed Elhoseiny, and Christian Claudel. Social-implicit: Rethinking trajectory prediction evaluation and the effectiveness of implicit maximum likelihood estimation. In *European Conference on Computer Vision*, pages 463–479. Springer, 2022.

[20] Nils-Bastian Heidenreich, Anja Schindler, and Stefan Sperlich. Bandwidth selection for kernel density estimation: a review of fully automatic selectors. *AStA Advances in Statistical Analysis*, 97:403–433, 2013.

[21] Allen Wang, Ashkan Jasour, and Brian C Williams. Non-gaussian chance-constrained trajectory planning for autonomous vehicles under agent uncertainty. *IEEE Robotics and Automation Letters*, 5(4):6041–6048, 2020.

[22] Lucas Janson, Edward Schmerling, and Marco Pavone. Monte Carlo motion planning for robot trajectory optimization under uncertainty. In *Robotics Research*, pages 343–361. Springer, 2018.

[23] Anna Mészáros, Julian F Schumann, Javier Alonso-Mora, Arkady Zgonnikov, and Jens Kober. Rome: Robust multi-modal density estimator. In *Proc. 33rd Int. Jt. Conf. on Artif. Intell*, 2024.

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

[25] Ruoqi Wei, Cesar Garcia, Ahmed El-Sayed, Viyaleta Peterson, and Ausif Mahmood. Variations in variational autoencoders-a comparative evaluation. *IEEE Access*, 8:153651–153670, 2020.

[26] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979, 2020.

[27] Donna Xu, Yaxin Shi, Ivor W Tsang, Yew-Soon Ong, Chen Gong, and Xiaobo Shen. Survey on multi-output learning. *IEEE Transactions on Neural Networks and Learning Systems*, 31(7):2409–2429, 2019.

[28] Sajjad Mozaffari, Omar Y Al-Jarrah, Mehrdad Dianati, Paul Jennings, and Alexandros Mouzakitis. Deep learning-based vehicle behavior prediction for autonomous driving applications: A review. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):33–47, 2020.

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

[30] Marc Peter Deisenroth, A Aldo Faisal, and Cheng Soon Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020.

[31] Don O Loftsgaarden and Charles P Quesenberry. A nonparametric estimate of a multivariate density function. *The Annals of Mathematical Statistics*, 36(3):1049–1051, 1965.

[32] Pascal Vincent and Yoshua Bengio. Manifold parzen windows. *Advances in Neural Information Processing Systems*, 15, 2002.

[33] Thomas Nagler and Claudia Czado. Evading the curse of dimensionality in nonparametric density estimation with simplified vine copulas. *Journal of Multivariate Analysis*, 151:69–89, 2016.

[34] Junwei Liang, Lu Jiang, Kevin Murphy, Ting Yu, and Alexander Hauptmann. The garden of forking paths: Towards multi-future trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10508–10518, 2020.

[35] Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[36] Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 1998.

[37] Xiaoxia Wang, Peter Tino, Mark A Fardal, Somak Raychaudhury, and Arif Babul. Fast parzen window density estimator. In *2009 International Joint Conference on Neural Networks*, pages 3267–3274. IEEE, 2009.

[38] Jia-Xing Gao, Da-Quan Jiang, and Min-Ping Qian. Adaptive manifold density estimation. *Journal of Statistical Computation and Simulation*, 92(11):2317–2331, 2022.

[39] Geoffrey J McLachlan and Suren Rathnayake. On the number of components in a gaussian mixture model. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(5):341–355, 2014.

[40] Harry Joe. *Dependence modeling with copulas*. CRC press, 2014.

[41] Yves I Ngounou Bakam and Denys Pommeret. Nonparametric estimation of copulas and copula densities by orthogonal projections. *Econometrics and Statistics*, 2023.

[42] Håkon Otneim and Dag Tjøstheim. The locally gaussian density estimator for multivariate data. *Statistics and Computing*, 27:1595–1616, 2017.

[43] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: Ordering points to identify the clustering structure. *ACM Sigmod Record*, 28(2):49–60, 1999.

[44] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, volume 96, pages 226–231, 1996.

[45] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[46] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.

[47] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[48] Cédric Villani. *Optimal transport: Old and new*, volume 338. Springer, 2009.

[49] Tarald O Kvalseth. Generalized divergence and gibbs' inequality. In *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, volume 2, pages 1797–1801. IEEE, 1997.

[50] Davoud Moulavi, Pablo A Jaskowiak, Ricardo JGB Campello, Arthur Zimek, and Jörg Sander. Density-based clustering validation. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 839–847. SIAM, 2014.

[51] Anna Mészáros, Julian F Schumann, Javier Alonso-Mora, Arkady Zgonnikov, and Jens Kober. Trajflow: Learning distributions over trajectories for human behavior prediction. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 184–191. IEEE, 2024.

[52] Jaspreet Singh Brar and Brian Caulfield. Impact of autonomous vehicles on pedestrians' safety. In *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017.

[53] Jonas Meyer, Henrik Becker, Patrick M Bösch, and Kay W Axhausen. Autonomous vehicles: The next jump in accessibilities? *Research in Transportation Economics*, 62:80–91, 2017.

[54] Jelena Pisarov and Gyula Mester. The future of autonomous vehicles. *FME Transactions*, 49(1):29–35, 2021.

[55] Michael Milford, Sam Anthony, and Walter Scheirer. Self-driving vehicles: Key technical challenges and progress off the road. *IEEE Potentials*, 39(1):37–45, 2019.

[56] Barry Brown, Mathias Broth, and Erik Vinkhuyzen. The halting problem: Video analysis of self-driving cars in traffic. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2023.

[57] Julian F Schumann, Jens Kober, and Arkady Zgonnikov. Benchmarking behavior prediction models in gap acceptance scenarios. *IEEE Transactions on Intelligent Vehicles*, 2023.

[58] Balakrishnan Varadarajan, Ahmed Hefny, Avikalp Srivastava, Khaled S Refaat, Nigamaa Nayakanti, Andre Cornman, Kan Chen, Bertrand Douillard, Chi Pang Lam, Dragomir Anguelov, et al. MultiPath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *International Conference on Robotics and Automation*, 2022.

[59] Kaouther Messaoud, Nachiket Deo, Mohan M Trivedi, and Fawzi Nashashibi. Trajectory prediction for autonomous driving based on multi-head attention with joint agent-map representation. In *IEEE Intelligent Vehicles Symposium*, 2021.

[60] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially acceptable trajectories with generative adversarial networks. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[61] Javad Amirian, Jean-Bernard Hayet, and Julien Pettré. Social ways: Learning multi-modal distributions of pedestrian trajectories with GANs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[62] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *European Conference on Computer Vision*, 2020.

[63] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. AgentFormer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[64] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *European Conference on Computer Vision*, 2020.

[65] Tianpei Gu, Guangyi Chen, Junlong Li, Chunze Lin, Yongming Rao, Jie Zhou, and Jiwen Lu. Stochastic trajectory prediction via motion indeterminacy diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[66] Bruno Ferreira de Brito, Hai Zhu, Wei Pan, and Javier Alonso-Mora. Social-VRNN: One-shot multi-modal trajectory prediction for interacting pedestrians. In *Conference on Robot Learning*, 2021.

[67] Esteban G Tabak and Eric Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.

[68] Esteban G Tabak and Cristina V Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

[69] Nicholas Rhinehart, Kris M Kitani, and Paul Vernaza. R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[70] Nicholas Rhinehart, Rowan McAllister, Kris Kitani, and Sergey Levine. Precog: Prediction conditioned on goals in visual multi-agent settings. In *Proceedings of the IEEE/CVF Int. Conf. on Comput. Vis.*, 2019.

[71] Apratim Bhattacharyya, Christoph-Nikolas Straehle, Mario Fritz, and Bernt Schiele. Haar wavelet based block autoregressive flows for trajectories. In *DAGM German Conference on Pattern Recognition*, 2020.

[72] Jianhua Sun, Zehao Wang, Jiefeng Li, and Cewu Lu. Unified and fast human trajectory prediction via conditionally parameterized normalizing flow. *IEEE Robotics and Automation Letters*, 7(2):842–849, 2021.

[73] Christoph Schöller and Alois Knoll. FloMo: Tractable motion prediction with normalizing flows. In *Int. Conf. Intell. Robot. Syst.*, 2021.

[74] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, 2009.

[75] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer Graphics Forum*, volume 26, 2007.

[76] Robert Krajewski, Tobias Moers, Julian Bock, Lennart Vater, and Lutz Eckstein. The rounD dataset: A drone dataset of road user trajectories at roundabouts in Germany. In *IEEE 23rd International Conference on Intelligent Transportation Systems*, 2020.

[77] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[78] George Papamakarios, Eric T Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22(57):1–64, 2021.

[79] You Lu and Bert Huang. Structured output learning with conditional generative flows. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[80] Olger Siebinga, Arkady Zgonnikov, and David Abbink. Uncovering variability in human driving behavior through automatic extraction of similar traffic scenes from large naturalistic datasets. In *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 4790–4796, 2023.

[81] Zhangjie Cao, Erdem Biyik, Guy Rosman, and Dorsa Sadigh. Leveraging smooth attention prior for multi-agent trajectory prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, 2022.

[82] Qingyuan Song, Wen Wang, Weiping Fu, Yuan Sun, Denggui Wang, and Zhiqiang Gao. Research on quantum cognition in autonomous driving. *Scientific reports*, 12(1):300, 2022.

[83] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

[84] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[85] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4):1–39, 2023.

[86] Anna Mészáros, Julian F. Schumann, Javier Alonso-Mora, Arkady Zgonnikov, and Jens Kober. Rome: Robust multi-modal density estimator. *Proceedings of the 33rd International Joint Conference on Artificial Intelligence*, 2024, in press.

[87] Julian F Schumann, Aravinda Ramakrishnan Srinivasan, Jens Kober, Gustav Markkula, and Arkady Zgonnikov. Using models based on cognitive theory to predict human behavior in traffic: A case study. In *2023 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2023.

[88] Frederik SB Westerhout, Julian F Schumann, and Arkady Zgonnikov. Smooth-Trajectron++: Augmenting the Trajectron++ behaviour prediction model with smooth attention. In *2023 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2023.

[89] Nicholas Rhinehart, Jeff He, Charles Packer, Matthew A Wright, Rowan McAllister, Joseph E Gonzalez, and Sergey Levine. Contingencies from observations: Tractable contingency planning with learned behavior models. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13663–13669. IEEE, 2021.

[90] Anna Mészáros, Javier Alonso-Mora, and Jens Kober. Studying the effect of explicit interaction representations on learning scene-level distributions of human trajectories. In *2026 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2026.

[91] Wenjie Luo, Cheol Park, Andre Cornman, Benjamin Sapp, and Dragomir Anguelov. Jfp: Joint future prediction with interactive multi-agent modeling for autonomous driving. In *Conference on Robot Learning*, pages 1457–1467. PMLR, 2023.

[92] Görkay Aydemir, Adil Kaan Akan, and Fatma Güney. Adapt: Efficient multi-agent trajectory prediction with adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8295–8305, 2023.

[93] Roger Girgis, Florian Golemo, Felipe Codevilla, Martin Weiss, Jim Aldon D'Souza, Samira Ebrahimi Kahou, Felix Heide, and Christopher Pal. Latent variable sequential set transformers for joint multi-agent motion prediction. *arXiv preprint arXiv:2104.00563*, 2021.

[94] Luke Rowe, Martin Ethier, Eli-Henry Dykhne, and Krzysztof Czarnecki. Fjmp: Factorized joint multi-agent motion prediction over learned directed acyclic interaction graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13745–13755, 2023.

[95] Djamel Eddine Benrachou, Sebastien Glaser, Mohammed Elhenawy, and Andry Rakotonirainy. Use of social interaction and intention to improve motion prediction within automated vehicle framework: A review. *IEEE Transactions on Intelligent Transportation Systems*, 23(12):22807–22837, 2022.

[96] Linhui Li, Jiazheng Su, Lihong Qiu, Jing Lian, and Ge Guo. Efin-mp: Explicit future interaction network for motion prediction. *IEEE Transactions on Intelligent Vehicles*, 2024.

[97] Xiaobo Chen, Fengbo Luo, Feng Zhao, and Qiaolin Ye. Goal-guided and interaction-aware state refinement graph attention network for multi-agent trajectory prediction. *IEEE Robotics and Automation Letters*, 9(1):57–64, 2023.

[98] Xiaoyu Mo, Zhiyu Huang, Yang Xing, and Chen Lv. Multi-agent trajectory prediction with heterogeneous edge-enhanced graph attention network. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):9554–9567, 2022.

[99] Yuxiao Chen, Boris Ivanovic, and Marco Pavone. Scept: Scene-consistent, policy-based trajectory predictions for planning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17103–17112, 2022.

[100] Vidyaa Krishnan Nivash and Ahmed H Qureshi. Simmf: Semantics-aware interactive multiagent motion forecasting for autonomous vehicle driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6813–6819. IEEE, 2024.

[101] Yuying Chen, Congcong Liu, Xiaodong Mei, Bertram E Shi, and Ming Liu. Hgcn-gjs: Hierarchical graph convolutional network with groupwise joint sampling for trajectory prediction. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13400–13405. IEEE, 2022.

[102] Qiao Sun, Xin Huang, Junru Gu, Brian C Williams, and Hang Zhao. M2i: From factored marginal trajectory prediction to interactive prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6543–6552, 2022.

[103] Prarthana Bhattacharyya, Chengjie Huang, and Krzysztof Czarnecki. Ssl-interactions: Pretext tasks for interactive trajectory prediction. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 1450–1457. IEEE, 2024.

[104] Ting Zhang, Mengyin Fu, Yi Yang, Wenjie Song, and Tong Liu. Edge-enriched graph transformer for multi-agent trajectory prediction with relative positional semantics. *IEEE Transactions on Instrumentation and Measurement*, 2024.

[105] Qifan Xue, Feng Yang, Shengyi Li, Xuanpeng Li, Guangyu Li, and Weigong Zhang. Rethink reynolds' rules: flock-inspired network for vehicle trajectory prediction. *The Journal of Supercomputing*, 81(6):802, 2025.

[106] Song Wen, Hao Wang, Di Liu, Qilong Zhangli, and Dimitris Metaxas. Second-order graph odes for multi-agent trajectory forecasting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 5101–5110, 2024.

[107] Miao Kang, Shengqi Wang, Sanping Zhou, Ke Ye, Jingjing Jiang, and Nanning Zheng. Ffinet: Future feedback interaction network for motion forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[108] Zhaobin Mo, Yongjie Fu, and Xuan Di. Pi-neugode: Physics-informed graph neural ordinary differential equations for spatiotemporal trajectory prediction. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pages 1418–1426, 2024.

[109] Dekai Zhu, Guangyao Zhai, Yan Di, Fabian Manhardt, Hendrik Berkemeyer, Tuan Tran, Nassir Navab, Federico Tombari, and Benjamin Busam. Ipcc-tp: Utilizing incremental pearson correlation coefficient for joint multi-agent trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5507–5516, 2023.

[110] Chenxin Xu, Maosen Li, Zhenyang Ni, Ya Zhang, and Siheng Chen. Groupnet: Multiscale hypergraph neural networks for trajectory prediction with relational reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6507, 2022.

[111] Frederik Diehl, Thomas Brunner, Michael Truong Le, and Alois Knoll. Graph neural networks for modelling traffic participant interaction. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 695–701. IEEE, 2019.

[112] Khaled A Mustafa, Daniel Jarne Ornia, Jens Kober, and Javier Alonso-Mora. Racp: Risk-aware contingency planning with multi-modal predictions. *IEEE Transactions on Intelligent Vehicles*, 2024.

[113] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*, 2023.

[114] Wei Zhan, Liting Sun, Di Wang, Haojie Shi, Aubrey Clausse, Maximilian Naumann, Julius Kummerle, Hendrik Konigshof, Christoph Stiller, Arnaud de La Fortelle, et al. Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. *arXiv preprint arXiv:1910.03088*, 2019.

[115] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 541–556. Springer, 2020.

[116] Julian F Schumann, Anna Mészáros, Jens Kober, and Arkady Zgonnikov. Step: Structured training and evaluation platform for benchmarking trajectory prediction models. *arXiv preprint arXiv:2509.14801*, 2025.

[117] Sarvesh Kolekar, Joost De Winter, and David Abbink. Human-like driving behaviour emerges from a risk-based driver model. *Nature communications*, 11(1):1–13, 2020.

[118] Olger Siebinga, Arkady Zgonnikov, and David A Abbink. Modelling communication-enabled traffic interactions. *Royal Society open science*, 10(5):230537, 2023.

[119] Maarten Hugenholtz, Anna Meszaros, Jens Kober, and Zlatan Ajanovic. Mode collapse happens: Evaluating critical interactions in joint trajectory prediction models. *arXiv preprint arXiv:2506.23164*, 2025.

[120] Kareem Othman. Exploring the implications of autonomous vehicles: A comprehensive review. *Innovative Infrastructure Solutions*, 7(2):165, 2022.

[121] Arda Kurt, John L Yester, Yutaka Mochizuki, and Ümit Özgüner. Hybrid-state driver/vehicle modelling, estimation and prediction. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 806–811. IEEE, 2010.

[122] Xin Huang, Guy Rosman, Igor Gilitschenski, Ashkan Jasour, Stephen G McGill, John J Leonard, and Brian C Williams. Hyper: Learned hybrid trajectory prediction via factored inference and adaptive sampling. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2906–2912. IEEE, 2022.

[123] Steffen Hagedorn, Marcel Hallgarten, Martin Stoll, and Alexandru Paul Condurache. The integration of prediction and planning in deep learning automated driving systems: A review. *IEEE Transactions on Intelligent Vehicles*, 2024.

[124] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Ben Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, et al. Tnt: Target-driven trajectory prediction. In *Conference on Robot Learning*, pages 895–904. PMLR, 2021.

[125] Junru Gu, Chen Sun, and Hang Zhao. Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15303–15312, 2021.

[126] Yuxiao Chen, Sander Tonkens, and Marco Pavone. Categorical traffic transformer: Interpretable and diverse behavior prediction with tokenized latent. *arXiv preprint arXiv:2311.18307*, 2023.

[127] Ye Yuan and Kris Kitani. Dlow: Diversifying latent flows for diverse human motion prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 346–364. Springer, 2020.

[128] Nachiket Deo and Mohan M Trivedi. Trajectory forecasts in unknown environments conditioned on grid-based plans. *arXiv preprint arXiv:2001.00735*, 2020.

[129] Zlatan Ajanovic, Bakir Lacevic, Barys Shyrokau, Michael Stolz, and Martin Horn. Search-based optimal motion planning for automated driving. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4523–4530. IEEE, 2018.

[130] Yuxiao Chen, Peter Karkus, Boris Ivanovic, Xinshuo Weng, and Marco Pavone. Tree-structured policy planning with learned behavior models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7902–7908. IEEE, 2023.

[131] Hendrik Berkemeyer, Riccardo Franceschini, Tuan Tran, Lin Che, and Gordon Pipa. Feasible and adaptive multimodal trajectory prediction with semantic maneuver fusion. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8530–8536. IEEE, 2021.

[132] Sumit Kumar, Yiming Gu, Jerrick Hoang, Galen Clark Haynes, and Micol Marchetti-Bowick. Interaction-based trajectory prediction over a hybrid traffic graph. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5530–5535. IEEE, 2021.

[133] Yuxiao Chen, Sushant Veer, Peter Karkus, and Marco Pavone. Interactive Joint Planning for Autonomous Vehicles. *IEEE Robotics and Automation Letters*, 9(2):987–994, February 2024. Conference Name: IEEE Robotics and Automation Letters.

[134] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719, 2021.

[135] Gustav Markkula, Ruth Madigan, Dimitris Nathanael, Evangelia Portouli, Yee Mun Lee, André Dietrich, Jac Billington, Anna Schieben, and Natasha Merat. Defining interactions: A conceptual framework for understanding interactive behaviour in human and automated road traffic. *Theoretical Issues in Ergonomics Science*, 21(6):728–752, 2020.

[136] Subhrajit Bhattacharya, Maxim Likhachev, and Vijay Kumar. Topological constraints in search-based robot path planning. *Autonomous Robots*, 33:273–290, 2012.

[137] Julius Ziegler and Christoph Stiller. Fast collision checking for intelligent vehicle motion planning. In *2010 IEEE intelligent vehicles symposium*, pages 518–522. IEEE, 2010.

[138] Ye Yuan. Khrylx/AgentFormer - https://github.com/khrylx/agentformer, May 2024. original-date: 2021-03-24T16:40:46Z.

[139] NVlabs/diffstack at CTT release - https://github.com/nvlabs/diffstack/tree/ctt_release.

[140] Phillip Karle, Maximilian Geisslinger, Johannes Betz, and Markus Lienkamp. Scenario understanding and motion prediction for autonomous vehicles—review and comparison. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):16962–16982, 2022.

[141] Ksander N De Winkel, Tugrul Irmak, Riender Happee, and Barys Shyrokau. Standards for passenger comfort in automated vehicles: Acceleration and jerk. *Applied Ergonomics*, 106:103881, 2023.

[142] Julian Bock, Robert Krajewski, Tobias Moers, Steffen Runde, Lennart Vater, and Lutz Eckstein. The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1929–1934. IEEE, 2020.

[143] Jiahe Chen, Jinkun Cao, Dahua Lin, Kris Kitani, and Jiangmiao Pang. Mgf: Mixed gaussian flow for diverse trajectory prediction. *Advances in Neural Information Processing Systems*, 37:57539–57563, 2024.

[144] Jing Zhao, Victor L Knoop, and Meng Wang. Microscopic traffic modeling inside intersections: Interactions between drivers. *Transportation science*, 57(1):135–155, 2023.

[145] Jing Zhao, Jairus Odawa Malenje, Yu Tang, and Yin Han. Gap acceptance probability model for pedestrians at unsignalized mid-block crosswalks based on logistic regression. *Accident Analysis & Prevention*, 129:76–83, 2019.

[146] Alessandro Corbetta, Jasper A Meeusen, Chung-min Lee, Roberto Benzi, and Federico Toschi. Physics-based modeling and data representation of pairwise interactions among pedestrians. *Physical review E*, 98(6):062310, 2018.

[147] Dongfang Yang, Keith Redmill, and Ümit Özgüner. A multi-state social force based framework for vehicle-pedestrian interaction in uncontrolled pedestrian crossing scenarios. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1807–1812. IEEE, 2020.

[148] Karim Fadhloun and Hesham Rakha. A novel vehicle dynamics and human behavior car-following model: Model development and preliminary testing. *International journal of transportation science and technology*, 9(1):14–28, 2020.

[149] Mohamed-Khalil Bouzidi, Christian Schlauch, Nicole Scheuerer, Yue Yao, Nadja Klein, Daniel Göhring, and Jörg Reichardt. Closing the loop: Motion prediction models beyond open-loop benchmarks. *arXiv preprint arXiv:2505.05638*, 2025.

[150] Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, et al. Towards learning-based planning: The nuplan benchmark for real-world autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 629–636. IEEE, 2024.

[151] Cole Gulino, Justin Fu, Wenjie Luo, George Tucker, Eli Bronstein, Yiren Lu, Jean Harb, Xinlei Pan, Yan Wang, Xiangyu Chen, et al. Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research. *Advances in Neural Information Processing Systems*, 36:7730–7742, 2023.

[152] Zeyu Mu, Fatemeh Jahedinia, and B Brian Park. Does the intelligent driver model adequately represent human drivers? In *VEHITS*, pages 113–121, 2023.

[153] Junctions `https://road-safety.transport.ec.europa.eu/european-road-safety-observatory/statistics-and-analysis-archive/roads/junctions_en`. Accessed: 2025-10-28.

[154] About intersection safety `https://highways.dot.gov/safety/intersection-safety/about`. Accessed: 2025-10-28.

Glossary

a2a – Agent to agent.
a2l – Agent to lane.
ADE – Average Displacement Error.
AF – AgentFormer.
AV – Autonomous Vehicle.

CCW – Counterclockwise.
CNN – Convolutional Neural Network.
CTT – Categorical Traffic Transformer.
CV – Constant Velocity.
CW – Clockwise.

DAG – Directed Acyclic Graph.
DBCV – Density-Based Cluster Validation.
DBSCAN – Density-Based Spatial Clustering of Applications with Noise.

ECE – Expected Calibration Error.

FDE – Final Displacement Error.
FJMP – Factorized Joint Motion Prediction.
FM – Flomo.

GAN – Generative Adversarial Network.
GCN – Graph Convolutional Network.
GMM – Gaussian Mixture Model.
GMoP – Graph-based Motion Prediction.
GNN – Graph Neural Network.
GRU – Gated Recurrent Unit.
GT – Ground Truth.

JSD – Jensen-Shannon Divergence.

KDE – Kernel Density Estimation.
kNN – k-Nearest Neighbors.

MID – Motion Indeterminancy Diffusion.

ML – Most Likely.
MPW – Manifold Parzen Windows.
MR – Miss Rate.
MSE – Mean Squared Error.
MTP – Multimodal Trajectory Prediction.

NF – Normalizing Flow.
NLL – Negative Log-Likelihood.

OPTICS – Ordering Points To Identify the Clustering Structure.

PCA – Principal Component Analysis.
PDF – Probability Density Function.
PECNet – Predicted Endpoint Conditioned Network.
PS – Path-Sharing.

RNN – Recurrent Neural Network.
RNN-AE – Recurrent Neural Network Auto-Encoder.
ROME – RObust Multi-modal density Estimator.

SM – Scene Modes.

T++ – Trajectron++.
TF – TrajFlow.

VAE – Variational Auto-Encoder.
VC – Vine Copulas.
VTP – Vehicle Trajectory Prediction.

WOMD – Waymo Open Motion Dataset.

Curriculum Vitæ

Anna Mészáros

Education

2022-2026	PhD at Cognitive Robotics Delft University of Technology, Netherlands. *Thesis:* Probabilistic Trajectory Prediction for Urban Driving.
2019-2021	MSc in Mechanical Engineering, *Cum Laude* Delft University of Technology, Netherlands. *Thesis:* Teaching Robots to Grasp Like Humans: An Interactive Approach
2016-2019	BSc in Robotics and Autonomous Systems University of Lübeck, Germany. *Thesis:* Learning the Relationship between Robot Geometries and Dexterous Workspaces using Artificial Neural Networks

Experience

2022-2024	MSc Thesis Supervisor Delft University of Technology, Netherlands Supervised three MSc students for their final thesis: Paul Féry, Tom Weinans, and Maarten Hugenholtz.
2023-2025	Teaching Assistant Delft University of Technology, Netherlands *MSc Course:* Machine Learning for Robotics (RO47002)
2018-2018	Data Scientist Intern CGI, Hamburg, Germany

List of Publications

Journals

1. **A. Mészáros**, G. Franzese, and J. Kober. "Learning to pick at non-zero-velocity from interactive demonstrations." IEEE Robotics and Automation Letters (RA-L), April 2022.

Conference Proceedings

1. **A. Mészáros**, J. Alonso-Mora, and J. Kober. "Studying the Effect of Explicit Interaction Representations on Learning Scene-level Distributions of Human Trajectories." To appear in 2026 IEEE Intelligent Vehicles Symposium (IV), Detroit, MI, United States, 2026.
2. **A. Mészáros***, J. F. Schumann*, J. Alonso-Mora, A. Zgonnikov, and J. Kober. "Rome: Robust multi-modal density estimator." Proceedings 33rd Iternational Joint Conference on Artificial Intelligence (IJCAI), Jeju, South Korea, 2024.
3. **A. Mészáros**, J. F. Schumann, J. Alonso-Mora, A. Zgonnikov, and J. Kober. "TrajFlow: Learning Distributions over Trajectories for Human Behavior Prediction." 2024 IEEE Intelligent Vehicles Symposium (IV), Jeju, South Korea, 2024.
4. G. Franzese, **A. Mészáros**, L. Peternel, and J. Kober. "ILoSA: Interactive learning of stiffness and attractors." 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2021.

Workshop Papers

1. **A. Mészáros**, J. Alonso-Mora, and J. Kober. "TrajFlow: Learning the Distribution over Trajectories." Long-term Human Motion Prediction Workshop - Workshop at ICRA 2023.

Under Review

1. M. Hugenholtz, **A. Mészáros**, J. Kober, Z. Ajanovic. “Mode Collapse Happens: Evaluating Critical Interactions in Joint Trajectory Prediction Models”.
2. J. F. Schumann*, **A. Mészáros***, J. Kober, A. Zgonnikov. “STEP: Structured Training and Evaluation Platform for benchmarking trajectory prediction models”.

Included in this thesis.