Action and trajectory prediction for Autonomous Driving

Laura Calem

HESAM UNIVERSITÉ

le cnam

ÉCOLE DOCTORALE SCIENCES DES MÉTIERS DE L'INGÉNIEUR
Centre d'études et de recherche en informatique et communications

THÈSE

présentée par : **Laura Calem**

soutenue le : **28 Mai 2024**

pour obtenir le grade de : **Docteur d'HESAM Université**

préparée au : **Conservatoire national des arts et métiers**

Discipline : **Mathématiques, Informatique et Systèmes**

Spécialité : **Informatique**

Action and trajectory prediction for Autonomous Driving

THÈSE dirigée par :
[**M. THOME Nicolas**] Professeur, ISIR, Sorbonne université

et co-encadrée par :
[**M. PÉREZ Patrick**] Directeur scientifique, valeo.ai

Jury

M. David PICARD	Directeur de recherche, École des Ponts Paris Tech	Rapporteur
M. Stéphane CANU	Professeur, Université Rouen Normandie	Rapporteur
M^{me} Catherine ACHARD	Professeure, Sorbonne Université	Examinatrice
M. Romain TAVENARD	Professeur, Université de Rennes	Examinateur
M. Patrick PÉREZ	Directeur scientifique, valeo.ai	Co-directeur de thèse
M. Nicolas THOME	Professeur, ISIR, Sorbonne Université	Directeur de thèse

THÈSE

Abstract

This PhD thesis, in the applicative context of autonomous driving, focuses on the exploration of diversity promoting mechanisms in generative models, which generate a probabilistic distribution of future trajectories given past trajectories. As trajectory forecasting datasets only provide one ground truth trajectory for a given past trajectory and scene spatial layout, many existing methods focus on the accuracy of the best predicted trajectory with respect to the ground truth trajectory. We aim to expand these methods by improving the intrinsic diversity of the predicted distribution, through the creation of a diversity-aware sampling mechanism that replaces traditional sequential sampling from generative models such as variational autoencoders (VAEs). We provide a way to generate samples according to the diversity exhibited in the training dataset, not only centered around the majority mode. The improvement of diversity, validated on nuScenes through a comprehensive set of metrics, is interesting with regard to the safety and smoothness of the planning operation, subsequent to trajectory forecasting. Furthering the diversity aspect in rare but safety-critical scenarios, we ask ourselves the question of expressing the diversity of events that are possible but yet unrepresented in the training dataset. This line of questioning raises the exploration of a much more challenging aspect: discovery. In order to generate a distribution that contains modes not present in the training dataset, we must carefully grow the training distribution according to an external admissibility function. The delicate balance between allowing the decoder of a generative model to generate from unknown latent codes and the necessity of generating admissible samples is explored in the second part of this thesis, with interesting results on a toy dataset.

Keywords: Variational Autoencoders, Trajectory Forecasting, Diversity, Discovery.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

The current artificial intelligence (AI) cycle, fueled by many successful commercial applications like chatGPT, Bard, Copilot, Dall-E and many more, is drawing a lot of public attention to the field. The growing public commentary alternatively focuses on the dangers of this type of technology on global employment, or the help these tools can provide in synergy with humans. But these applications are still mostly confined in the immaterial space of data centers and web interfaces. When we close our laptops and phones, AI becomes less ubiquitous.

This lack of penetration to the material world is due to the many constraints it poses, which can be regrouped into engineering challenges and safety concerns. First, interfacing programs with the real world is a significant engineering challenge. Even interfacing programs with the internet is an engineering challenge, one needs to create a crawler to get information from the internet, then a parser to extract information, for example. All these software blocks are oftentimes invisible to the outside of a system, but usually make up the most of a software system. Interfacing with hardware is of similar complexity, if not higher, as many physical systems weren't initially designed with any software component to begin with. Even when hardware components, such as LiDAR (Light Detection And Ranging) sensors, come with software drivers, integrating from the raw data to usable format for deep learning models is a whole engineering hurdle in itself. Second, physical systems from automated doors to autonomous driving systems have the potential to cause physical harm to humans. Even putting aside considerations such as public perception influencing regulation, safety requirements for autonomous systems are trying to move as fast as the technology itself, which adds overhead to the integration of automated systems into the material world.

1.1.1 Autonomous Driving Assistance Systems

From the European's Prometheus project in 1987 [1], marking the start of driving (figure 1.1 left), to today's commercial highly assisted driving systems like Tesla's "autopilot" or Mercedes-Benz Drive Pilot, autonomous driving research has come a long way, pushed by public challenge goalposts like the DARPA challenge in 2004 and 2007 and major demoed advancements like the Google Car in 2009.

Figure 1.1: **Autonomous driving systems historical landmarks**. (left) the interior of a Mercedes W140 S-Class, re-engineered in the context of the Prometheus project, which drove 1,678 kilometers from Munich to Copenhagen with minor human intervention. (right) A Cruise autonomous robotaxi in the streets of San Francisco, June 2022.

Self driving cars, under the current denomination Autonomous Driving Assistance Systems (ADAS), are the pioneering application of artificial intelligence systems intertwined with engineering constraints and challenge to relieve or enhance human behavior. The automotive industry, which can be considered mature, has embraced the capabilities of modern computer vision and gave rise to a number of major players, which interestingly are not traditional automakers: Baidu, Tesla, Waymo or Yandex have created programs to make use of ADAS in commercial cars and products.

While the terms commonly employed to describe ADAS, like "autonomous vehicle" or "self-driving car" are useful and used in the common discourse, when talking about implementation and actual systems more distinctions need to be made in order to have a common understanding about driving automation capabilities of a vehicle. The Society of Automotive Engineers (SAE), proposed in 2016 a classification of different levels of driving automation (figure 1.2), ranging from "Level 0" where there is no autonomous capabilities besides enhanced warnings to the driver or automatic emergency braking systems, to "Level 5", where a vehicle is fully autonomous and capable or self-driving without supervision (to the point where the steering wheel is optional).

The SAE classification system is not without flaws, as any standard. One notable issue is the ambiguity that emerges between the system's intended purpose and its actual application in practice.

[1] https://web.archive.org/web/20180814201633/http://www.eurekanetwork.org/project/id/45

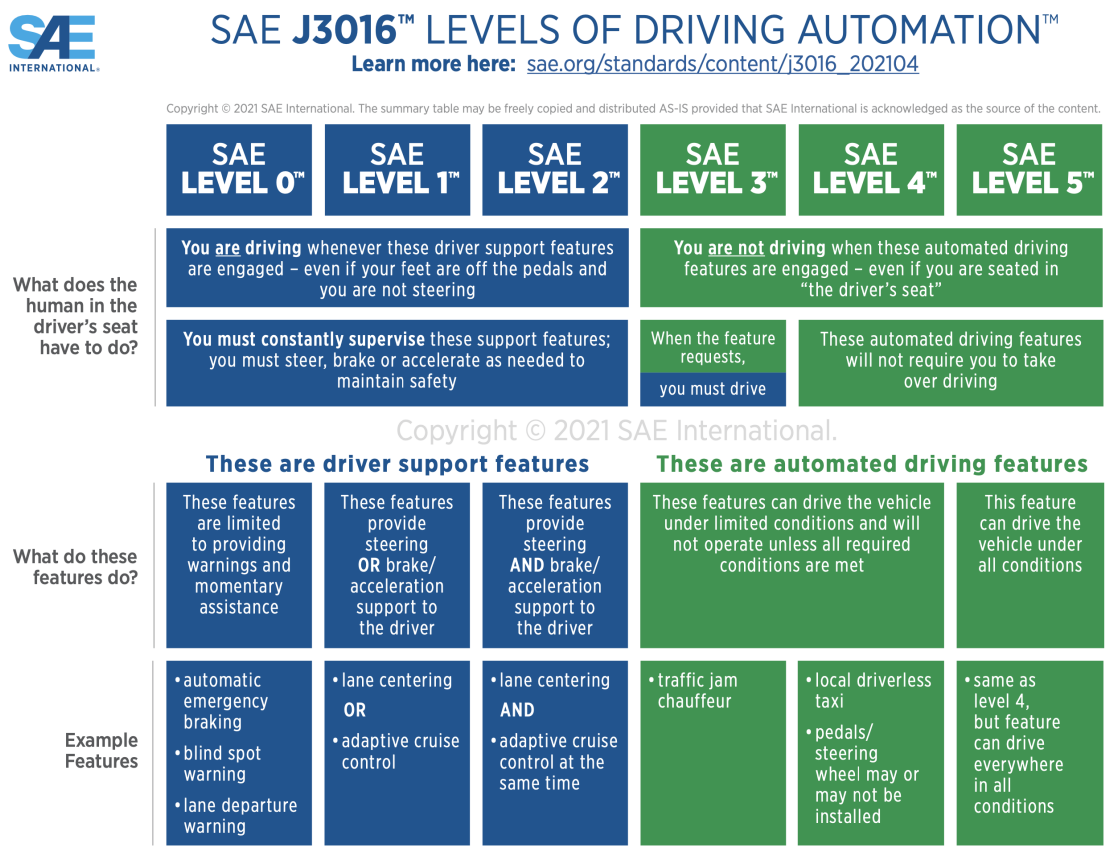


Figure 1.2: **Classifications of the different levels of driving automation**. The taxonomy proposed by the Society of Automotive Engineers (SAE) aims to provide common terms to designate autonomous capabilities.

This discrepancy becomes increasingly apparent as numerous systems claim compliance with "Level 3" criteria. The classification system aims at defining the distribution of responsibility between the user and the autonomous system. While it is often thought as merely a list of features the system must implement. For level 3, the boundary between the self-driving system and the driver shifts as for many features, the self driving operation must be supervised at all times by the driver, with constant attention. As highlighted in works like [Bauchwitz and Cummings, 2020], humans typically struggle at maintaining attention without active engagement, which for driving systems opens a new set of regulatory, safety and legal accountability concerns.

Despite these flaws, the classification system has started gaining traction in the general discourse and can be used as a levelled reference point of comparison between different commercial systems and their marketed names. Some level 3 systems are starting to be available to the general public, like Mercedes-Benz' Drive Pilot and BMW Personal Pilot L3, where "L3" in the name refers to Level 3. These systems are examples of first commercial applications of level 3 systems.

Before diving into how deep learning enables autonomous driving and in which way the present thesis can further understanding and reliability of such systems, a word needs to be said on the task of autonomous driving in itself, in a world where climate change is a life-threatening issue. Shifting systematic use of cars for personal transportation to more energy-efficient modes (by reducing the need of transportation, mode of transportation or the energy cost of operating cars) is a desirable path towards sustainable transportation, but individual transportation will likely remain part of the transport infrastructure. Being independent from whether the underlying car is electric or not, autonomous systems may have their importance in such a world, where driving could have a reduced importance in the daily lives of people and could be delegated to robots, in order to avoid the inconvenience of learning how to drive, buying and maintaining a personal car (autonomous robotaxis could fill specific needs), or to avoid the unpleasant experience of driving in heavy traffic or through cities.

1.1.2 Roles and tasks of Deep Learning for Autonomous Driving

These systems rely on a stack of technologies to run, many of which are deep learning based. An array of sensors including an array of cameras giving a 360° view of the surroundings and LiDAR (Light Detection and Ranging) typically provide the source data [Varghese et al., 2015], which are processed by deep learning models. Sometimes RADAR (lexicalisation of Radio Detection and Ranging) is also present in the sensor suite, most notably to overcome the weaknesses of LiDAR in adverse weather conditions. Table 1.1 breaks down the main characteristics of the main sensors. These models often include convolutional neural networks (CNNs) and transformers to understand the environment exposed by RGB images or LiDAR point clouds in real time and detect lanes, other vehicles, pedestrians, bicycles, etc. [Janai et al., 2020].

-	Camera	LiDAR	Radar
Range	$<$ 100m	150m	$>$ 200m
Resolution	good	poor	average
Adverse conditions	poor	average	good
Poor lighting conditions	poor	good	good

Table 1.1: **Typical sensor suite in autonomous vehicles**. The overview of the primary characteristics of the various sensors highlights their complementarity.

Perception tasks As the basis of the autonomous system pipeline, the quality of the perception stack determines a large portion of the outcomes performance. The synergy between cameras and LiDARs

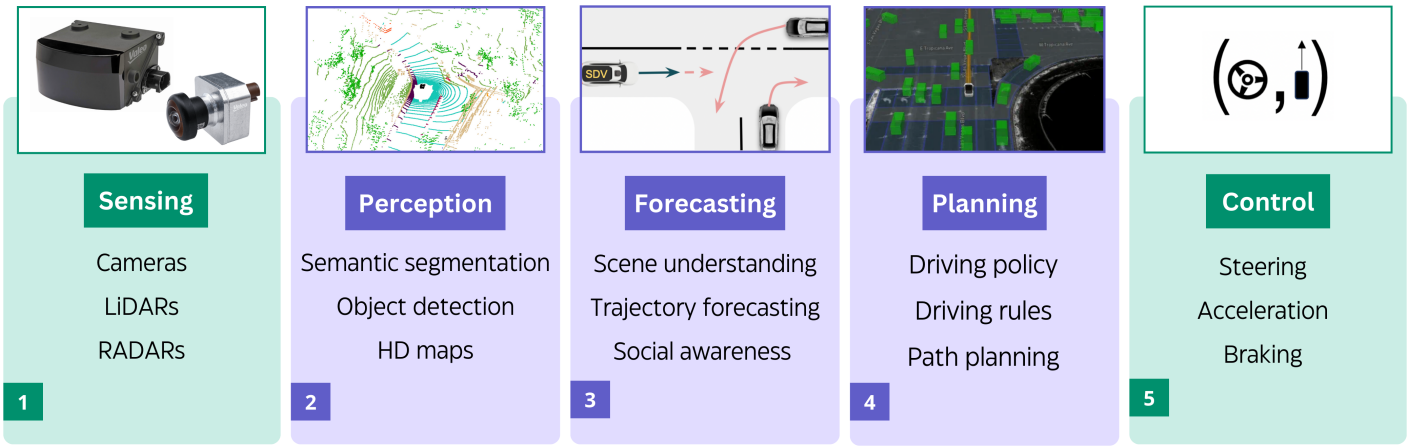


Figure 1.3: **Typical autonomous driving system deep learning pipeline**. End-to-end autonomous driving systems can be decomposed in several critical steps, each of which can be an independent area of research. Leftmost of this chart is perception of the surroundings, with raw sensory inputs being processed by deep learning systems and made into a standardized map via detection and segmentation. This representation is the basis for the more abstract reasoning that needs to occur in order to forecast the evolution of the dynamical scene including all moving and non-moving agents; this, in turn, is the basis for ego-vehicle planning. The planning step is then translated in instructions to control the car's two possible axes of movement: acceleration / braking and steering of the wheel. In green are the physical steps, purple ones are those requiring (deep learning) processing.

allows for some redundancy and complementarity and can be leveraged via sensor fusion to improve object segmentation and detection of critical road elements [Bai et al., 2022, Liu et al., 2023].

As an established leader in automotive parts including sensors, Valeo, which funded this work, is interested in the research and development of autonomous driving systems. With the creation of the Valeo.ai research laboratory in 2018, Valeo's renewed interest in artificial intelligence research aims to impact all levels of an autonomous vehicle's technological stack, from sensors with the commercially available SCALA LiDAR sensors, to advances in semantic segmentation and trajectory forecasting.

From sensors to detected road elements, this first stage of deep learning tasks, centered around perception, car readily be used in some applications: emergency braking systems, where an AI-based system is able to detect an impending obstacle and break accordingly. Lane detection can be used to automatically follow the current driving lane or warn the user when trajectory drifting occurs. Even though these systems do not form a fully autonomous driving system, they form the basis of a hardware and software suite that is used for safe driving and autonomous assistance to the driver.

Planning and driving tasks Downstream from the perception stack, the detected elements are used in various systems, with the ultimate goal of combining all components to produce a fully autonomous end to end driving pipeline capable of self driving a vehicle without human supervision, in compliance

with all safety and regulatory requirements.

At the heart of autonomous driving lie **prediction** and **planning**. While perception is of course the basis upon which everything else is built, the decision-making capabilities of an autonomous driving system are often decomposed into prediction and planning [Schmerling et al., 2018, Fan et al., 2018, Zeng et al., 2019a, Nishimura et al., 2023, Cui et al., 2021]. The prediction step analyses the outputs of the perception step, and produces future trajectories of surrounding agents in the scene, that span the range of possible scenarios, typically ranked by likelihood. These predictions then constitute the input of the planning step that is tasked with computing the ego-car trajectory.

From a successful planning, deterministic rules and engineered systems can then be used to compute the necessary steering and acceleration (positive or negative) values that are to be applied to the vehicle in order to realize the planned trajectory, a task commonly referred as **control**. These steps, perception, prediction, planning and control, form the main tasks for any end-to-end autonomous driving system, all of which rely heavily on deep learning methods.

1.2 Motivation: Trajectory prediction and safety

With 1.35 million fatalities in road accidents every year worldwide [2], there is certainly room for improvement in the domain of safety for cars and vehicles in general. An automated system provides a number of advantages over humans: it is always on, not subject to being tired, drunk or distracted the same way humans are. These characteristics make the use of autonomous driving systems appealing in many situations, as road transport for goods and people is still a cornerstone of modern societies. However, autonomous driving systems can also fail in unexpected ways compared to humans (see figure 1.4 [3]).

As with any technology, artificial intelligence or not, human safety should be a concern. When humans are involved, we can often rely on a certain degree of predictability of human behavior to craft safety and regulation systems around. When we delegate some decision-making to an artificial intelligence system, we likewise should understand well the advantages and drawbacks of these solutions to ensure that safety isn't compromised. With the diluted liability of autonomous systems compared to human drivers, the safety component is even more critical, as injuries and casualties in crashes involving autonomous vehicles lead to more complex (for now) litigations.

As a complex system, an autonomous driving system can have multiple points of failure causing car accidents and crashes:

- **Perception Errors**: This involves the vehicle's sensors and algorithms failing to correctly perceive

[2] https://www.who.int/data/gho/data/themes/topics/topic-details/GHO/road-traffic-mortality
[3] https://www.washingtonpost.com/technology/interactive/2023/tesla-autopilot-crash-analysis/

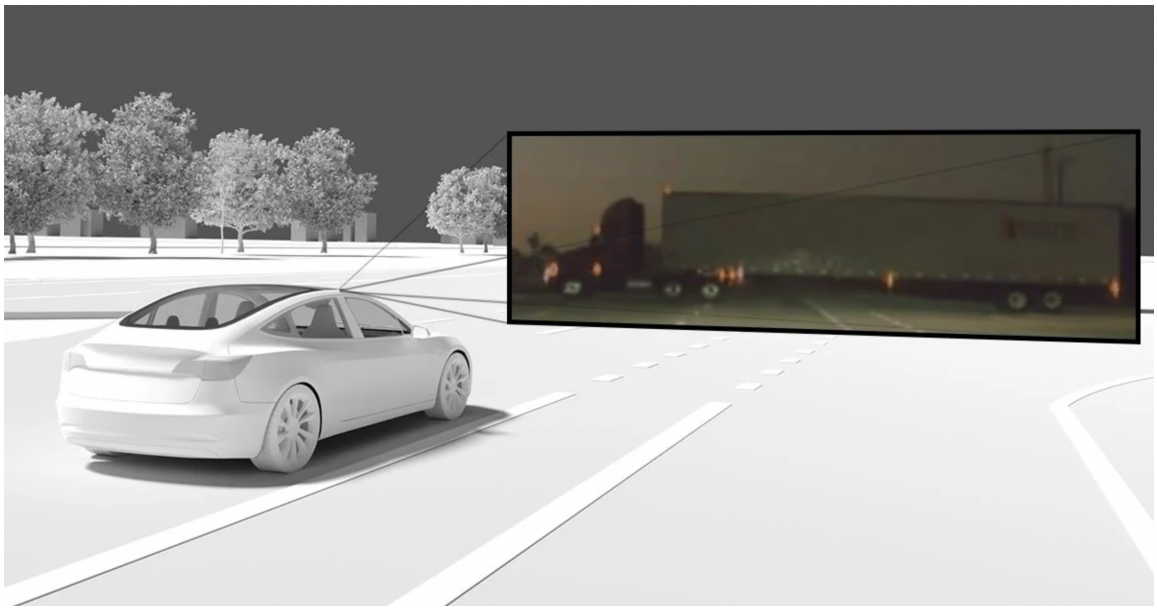


Figure 1.4: **Perception failure car crash**. In May 2019, failure to detect a semi-truck caused the car on Autopilot to crash into the truck at 110 km/h, instantly killing the driver. Neither the autonomous system nor the driver activated the brakes. Image source: Washington Post

and interpret the environment. This can be due to limitations in sensor technology, obstructed views, or unusual or unexpected scenarios that the system isn't trained to handle. For example, Tesla's Autopilot system has had incidents where it failed to recognize stationary objects or misinterpreted vehicle paths (see example in figure 1.4).

- **Trajectory Prediction Failures**: Algorithms to predict the movements of other vehicles, pedestrians, and objects can be inaccurate inaccurate, leading to crashes. This type of failure might occur if the system misjudges the speed or direction of another object, or fails to anticipate human error.

- **Software Bugs or Glitches**: Like any complex software system, autonomous driving systems can have bugs or glitches that may lead to incorrect actions or inactions. For example in April 2022, the crash of a TuSimple semi-truck highlighted that an outdated command was executed instead of being erased.

- **Hardware Failures**: Sensor malfunctions or mechanical problems can impair source data. If perception and prediction algorithms are not robust, it can lead to crashes.

- **Human Error**: In semi-autonomous vehicles, where human intervention is still a key component, driver inattention or incorrect actions can lead to crashes, a risk sometimes enhanced in systems

where the driver has to be attentive at all times but the system is doing the actual driving. Even in fully autonomous modes, human error in other vehicles can be a factor.

- **Adverse Weather Conditions**: Autonomous systems can struggle in heavy rain, snow, or fog, where sensor visibility is reduced. Notably, LiDAR's reliance on light waves that bounce back from obstacles is greatly impaired in the snow or rain as beams can bounce on the rain drops or snowflakes.
- **Complex Traffic Situations**: Some accidents have occurred in complex traffic situations that the autonomous system couldn't navigate safely, such as merging lanes or busy intersections.

All these points of failures are not to say autonomous systems are inherently more dangerous as human drivers, as we have seen with the number of fatalities worldwide that humans are not particularly safe either. When considering automation of large vehicles that can drive at considerable speeds in an open environment, it's a sign of healthy development to consider all possible failure cases in order to mitigate them.

In the relatively recent development of commercial autonomous vehicle technology that can operate on open road, proactive safety management is imperative for industry-wide credibility. The nascent regulatory frameworks governing this technology present a double-edged sword. On one hand, they allow for experimental application in real-world scenarios, accelerating the evolution of automotive automation at an unprecedented rate. On the other, this rapid progression necessitates a heightened emphasis on the security aspects of these systems, as an open road setting makes anyone, not only vehicle owners, at risk. This concern must be addressed by manufacturers and researchers at the earliest stages of development. Neglecting security considerations could result in a proliferation of accidents, eroding the reputation of individual brands and potentially triggering comprehensive bans on the technology. For instance, in October 2021, an incident involving a Pony.ai autonomous vehicle colliding with a road sign led to the suspension of the company's testing permit by the California Department of Motor Vehicles.

Given the security challenges and that errors in autonomous vehicle algorithms can directly impact the lives of people, safety should be in the hands of everyone involved in order to mitigate risks: regulatory bodies, automakers, self-driving systems developers and, upstream of all, research. As such, many works in autonomous driving consider safety as their primary focus. For example robustness is an increasingly relevant area of research applied to autonomous driving [McAllister et al., 2017, Corbière, 2022, Li et al., 2020]. The industrial and research context of the present thesis focuses on trajectory prediction, a key component of the automation pipeline downstream of perception, which guides planning. A better and especially a more diverse trajectory prediction enables the subsequent planner to be less over-cautious [Cui et al., 2021], which improves the overall safety of autonomous

systems.

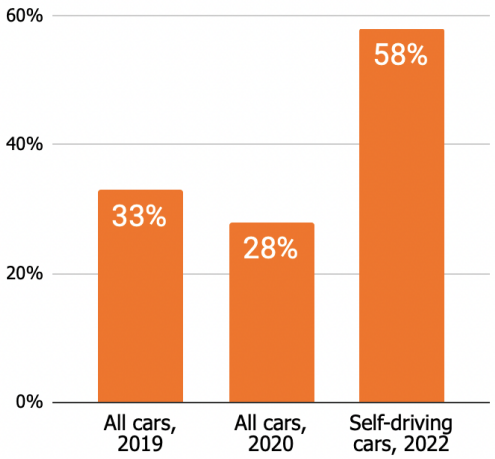
The following sections aim to introduce the topic of trajectory prediction in the context of autonomous driving, and highlight performance and security challenges in the area to contextualize the present thesis work.

1.2.1 Trajectory forecasting

In the global scheme of end-to-end autonomous driving (see figure 1.1), trajectory prediction plays a central role, as the basis of decision-making processes. Being the backbone of planning, it is critical for both safety and the smoothness of predicted trajectories. Even in non-fatal car crashes, erratic behavior from autonomous drivers is a major issues as other drivers cannot be expected to adjust their behaviors according to whether there is an autonomous system driving nearby cars or not. One common failure mode of an over-cautious planning system is braking in the middle of an otherwise safe trajectory, as evidenced by the high rate of rear-ended autonomous drivers (see figure 1.5, numbers sourced in [4]).

A higher percentage of rear-end non fatal crashes could just mean that more serious car crashes are less frequent in autonomous cars, which would be indicative of the safety of these systems rather than evidence for poor planning. Reliable data on autonomous driving car can be hard to come by because it's manufacturer dependent and the liability in the event of a crash is sometimes in a gray area. While this responsibility issue has been debated for a long time [Marchant and Lindor, 2012], crash attribution data can be hard to aggregate and compare to non autonomous car data, because of an array of issues impacting the comparability of data, like the differing legislation of the areas where autonomous vehicles are deployed, the level of automation of the cars involved in crashes, whether such systems were engaged at the moment of the crash, and the overall heterogeneity of the data [5]. Nevertheless, focusing only on the rear-end damage site of involved vehicles, with 58% for ADAS vehicles it is still higher than the 33% of non-ADAS

Figure 1.5: **Rear-end crash percentages**. Self-driving cars tend to be damaged mostly in the rear part of the car. Source: (U.S) NHTSA.



[4] https://www.nhtsa.gov/sites/nhtsa.gov/files/2022-06/ADS-SGO-Report-June-2022.pdf

[5] For example, in NHTSA's Summary Incident Report Data 2022 on aggregated crashes in ADAS systems, the first example involved a Lane Keep Assist system, activated on speeds above 37 mph, on a road where the speed limit is 25 mph

vehicles crash damage that impact the rear of the vehicle [6]. The phenomenon appears to be quite common as it bears a name: phantom braking, and could be indicative of poor planning, which can be due to errors in perception but also in trajectory forecasting. Planners are limited in their planning by the trajectory prediction step, and if the latter isn't good enough, additional failsafes can be put in place, which can avoid fatal crashes but also impair the smoothness of the autonomous vehicle's planning. If the vehicle trajectory is unreliable and behaves too differently from a human driver, the overall traffic could be negatively impacted by the presence of autonomous vehicles. A better trajectory prediction can and does mean a better planning, especially because planners can be overly cautious [Zhan et al., 2016, Cui et al., 2021, Tas et al., 2018], which could be a source of rear-end crashes in autonomous vehicles. One way to mitigate this problem is to have a trajectory forecasting step that is better aware of all the possibilities that can happen in the scene, and thus the issue of correctly representing the diversity of the potential future trajectories, even if they represent a minority mode, has been deemed of interest in the present thesis subject.

In summary, in the context of autonomous driving safety, trajectory prediction is a crucial aspect and working on the diversity of the predictions to improve the smoothness, safety and human-like predictability of autonomous driving agents on the road is an important aspect of trajectory prediction.

1.2.2 Simulators

Simulators like CARLA [Dosovitskiy et al., 2017] are invaluable in developing autonomous driving systems, as they enable testing in a closed-loop manner. Closed-loop testing refers to a dynamic process where the system is continually influenced by the feedback it receives, mimicking real-life driving scenarios. This approach contrasts with open-loop testing, which is more static and based on a real-world dataset. In open-loop testing, the system is evaluated against a fixed set of data, typically comprising just one trajectory (the ground truth acquired in the real-world dataset). This limitation necessitates open-loop validation, as the system cannot interact with or alter the dataset. By using simulators, models can be tested in closed loop, so that the testing conditions are closer to real-world operation while still being safer and easier to test that on an actual autonomous car on the road.

As such, simulators are a good testing step for models, between open-loop validation on real-world datasets and testing in the wild. However, this validation step is only as good as the simulator we use. If if doesn't respond properly to the vehicle's actions or doesn't exhibit complex enough behavior, this step could become ineffective at spotting potential issues for a model before it's deployed in the wild.

In this context, it's important to have simulators that can represent as faithfully as possible the actual distribution of behaviors encountered in the real world. Of course, it is not practically feasible

[6] https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812183

because if we had a way to perfectly model trajectories we wouldn't need models to predict them in the first place. However, from a simplistic rule-based simulator to a complex and probabilistic environment that can sometimes exhibit out-of-distribution situations to test the reliability of the autonomous system being tested, there is certainly room for improvement.

Increasing the availability and effectiveness of diversity mechanisms for trajectory prediction in general could allow for a better stochastic representation of the real world in simulators. By effectively modeling the training distribution including minority trajectories that rarely happen, a simulator could be able to more effectively test for adverse scenarios.

In summary, while simulators are useful in the development and safety of autonomous driving algorithms, being too reliable in their reactions hinders their effectiveness as testing tools. Furthering research in the diversity of trajectory prediction could also help the reliability of simulators.

1.2.3 Discovery

As most things in deep learning, trajectory prediction models mostly rely on extracting regularities from the data. Despite being easier to optimize for real-world datasets which only contain one ground truth future trajectory per past trajectory, predicting only one possible future trajectory proved too limited for the actual goals of autonomous driving, which include safety. As such, all current methods starting around [Lee et al., 2017] predict a probabilistic distribution for future trajectories. Predicting other trajectories that might be less likely to happen improves planning [Cui et al., 2021] and metrics have been developed to assess whether the proposed methods are still accurate with respect to the original real-world dataset.

However, as stated before, diversity might be key in the last safety percentage points that make a model go from good for public benchmarks to good for actual deployment in a self-driving car. Following Pareto's principle, the rarer trajectories are harder to gather in a dataset.

Rare occurrences might also be safety critical, like instances of crashes that cannot be reliably or ethically gathered in a dataset. As [Bansal et al., 2018] demonstrated, merely putting a penalty for collisions in a model training objective does not constitute an effective way of preventing a model from drifting its predicted trajectories towards a collision: if such collisions were not seen during training, the penalty was never applied.

This is where *discovery* could be a useful tool in the autonomous driving arsenal, representing an extreme yet distinct form of diversity. Defined as the generation of samples that bear no resemblance to the ones seen in the training data, discovery poses a new exciting challenge as it touches upon extrapolation. Relying on external sources of constraints in a way that helps enhancing the reliability of the end models is a powerful tool to improve autonomous driving technologies. Edge cases and

rare crash-inducing events require a lot of data to cover, some of which should be synthetic as no real car should be harmed during data acquisition. Carefully handcrafting edge-case scenarios in a simulator to improve safety of the final model, while not benefiting performance on public benchmarks, is a costly task that is often not prioritized. In this setting, a more systematic way of generating out-of-distribution trajectories that are admissible under admissibility constraints like the road layout or driving rules could be a valuable addition in crafting safety-critical models to be deployed in production.

Unlike classical algorithms that rely solely on training data for predictions, discovery demands a more nuanced method, one that can adeptly handle the complexities of extrapolation. This distinction underscores the fundamental difference between mere diversity and discovery, with the latter pushing the boundaries of what autonomous systems can predict and adapt to. It is also hard to test, as no public benchmark explicitly tackles this question. As such, the challenge that this thesis tackles is to assess whether discovery can be effectively developed for autonomous driving, in order to pave the way for more comprehensive systems.

As a more recent task, especially in probabilistic trajectory forecasting, discovery poses several key scientific challenges that we aim to address in this work, as no prior work exists yet to answer these: (1) devise an experimental setup that can effectively test for discovery of modalities that are admissible yet unseen in the training data, and (2) validate whether discovery is at all possible with an external admissibility function.

In summary, discovery is an interesting problem to tackle in the context of autonomous driving, as it can help with corner case reliability and safety. Finding a systematic way to discover and include rare but admissible events in the possible future generated trajectories of an agent is a step towards safer systems.

1.3 Contributions and outline

In this thesis, we place ourselves at the outskirts of trajectory prediction, by tackling the diversity and discovery problems.

The manuscript is organized as follows.

- **Chapter 2**: The next chapter lays out a general overview of the research areas most relevant for this thesis. The underlying task being probabilistic trajectory forecasting, we start by introducing the literature in relation to autonomous driving. As the main topics of the thesis are diversity and discovery, we then delve into these niches by presenting existing work in the area either directly related to trajectory forecasting when it exists (for diversity) or by drawing

links between our subject and works relevant for discovery.

- **Chapter 3: Diversity for trajectory forecasting**. In this chapter, we start by exploring ways of leveraging all the data present in the training dataset, in order to predict a more diverse future trajectory set spanning the possible futures. To this end, we present an elegant mathematical tool, Determinantal Point Processes (DPPs), that can model negative correlations and thus be useful in promoting the diversity of a generated set. After reviewing its use in current methods, we assess its performance on a real world dataset in order to identify its limitations, and correct them by presenting a novel model, DIVA, that builds upon an underlying generative model to bolster the diversity of the generated trajectory set. With DIVA, an effective way of combining admissibility and diversity is proposed, and extensively tested to validate this approach for real world trajectory prediction, along with the impact of each component on the performance of the overall model.

- **Chapter 4: Discovery in the absence of training data**. In this chapter, we explore further and ask ourselves what if there are modalities absent from the training dataset, and set proof-of-concept work for the generation of such modalities. Self labeling can be an effective way of expanding the generative distribution from the training distribution to the admissible distribution, which is the end goal of discovery. One of the main challenges in discovery is the creation and integration of a suitable admissibility function. However, this admissibility function has to contain the constraints of admissibility, but not the entire definition of the objects we want to generate, as it would (1) be too complex to create and (2) render useless the whole generation process. As the admissibility function contains necessary but not sufficient information for generating elements from the target distribution, it has to be combined with information contained in the training distribution. This fusion between the two is the core issue of discovery, and we propose in this chapter a method to balance the two in order to expand the training distribution towards the admissible distribution. We also derive an experimental setup in order to validate the model, along with extensive experiments to assess the performance of different ways to perform the self-supervised training scheme used for this method.

- **Chapter 5: Conclusion and perspectives**. Finally, we ask ourselves how principles for diversity and discovery can be improved, by exploring ways of systematically leveraging discovery. We also explore how the systems we developed for diversity and discovery in the context of trajectory forecasting could be used and improved in other contexts and tasks where external constraints could also be leveraged for improving the diversity of generated samples.

Chapter 2

Diversity in trajectory generation

Chapter Abstract

In this chapter, we propose a general overview of the topic of diversity in trajectory generation models, starting with the base task of probabilistic trajectory forecasting, central to the literature surrounding autonomous driving. In addition to an explanation of the usefulness and implications of diversity, it includes a definition of the problem at hand, a review of the literature and a discussion on how to adequately evaluate the subject. First explained in the applied context of trajectory forecasting, where diversity is useful for corner cases, safety and planning.

As a natural extension of the diversity issue, we touch upon the subject of discovery, a more exploratory concept with scarce existing literature in the context of trajectory forecasting. Other works in more fundamental contexts do exist and can be studied to define the task, which we detail in the last subsections of this chapter. We describe research attempts made in this field so far and create links to the trajectory generation topic of this thesis.

Contents

2.1 Trajectory forecasting

Downstream from the perception tasks that ultimately concur to produce a high fidelity HD-maps [Liu et al., 2020] (see figure 2.1 for an example), trajectory forecasting is a growing task of interest in the development of autonomous driving systems. While end-to-end methods that integrate both perception and trajectory prediction emerge [Bojarski et al., 2016, Casas et al., 2020, Chitta et al., 2021, Kendall et al., 2019], they require a comprehensive set of data to perform all the tasks of the pipeline and are typically very slow, even if several recent methods aim to make systems that work in real time [Casas et al., 2021, Li et al., 2022]. They will probably ultimately be the type of models that govern real-time planning in autonomous cars from input sensors only, but the current levels of automation that we see in commercial products still benefit greatly from the separation of the perception and trajectory forecasting tasks. As for the current applications, robotaxis (Zoox, Waymo, Cruise) can rely on pre-mapped HD-maps for the bulk of static road elements, and driving assisting products (Wayve, Tesla) provide guidance and interpretability that is based on having a distinct and intelligible perception stack output. Moreover, having the trajectory prediction part separated allows to leverage the large body of literature on perception and focus on the forecasting problem. Even as a stand-alone task, there are plenty of challenges to solve for trajectory prediction to be usable as a part of commercial products that provide automation to vehicles on the road. The main task is trajectory prediction, as in being able to accurately predict the future trajectory of an agent given its past trajectory.

2.1.1 Task

The trajectory prediction task refers to the forecasting of other agent's future actions in the scene. Agents are defined as anything dynamical: pedestrians, bicycles, motorcycles, cars and other larger vehicles. As there are often multiple agents in any given scene, the problem is also referred as "multi agent forecasting". It is by no means a new problem. In a context broader than autonomous vehicles, Kalman filters [Kalman, 1960] or Gaussian Process models [Williams, 1998, Wang et al., 2007] were famous tools for multi agent forecasting problems in the "pre-deep learning" era. Since then, many deep learning based approaches have been developed [Alahi et al., 2016, Lee et al., 2017, Wang et al., 2018, Deo and Trivedi, 2018, Sadeghian et al., 2019, Salzmann et al., 2020, Roddenberry et al., 2021, Gilles et al., 2022, Pang et al., 2023], in order to improve the trajectory forecasting accuracy on the trajectories recorded from the real-world datasets that are commonly used.

2.1.2 Input data

Considering the trajectory forecasting task as separated from the perception task offers a number of advantages. Aside for not having to compute any HD-map from sensor data in a more complex unified model, the trajectory forecasting step being separate means more leeway to choose input data.

The most common type of input data is a rasterized image made by adding all modalities together on a single RGB image (see figure 2.1 left). Datasets usually expose different modalities in separate layers (pedestrian crossings, drivable area layout, agent trajectories, traffic lights, etc.), which are then rasterized into a single image-like grid and fed as a traditional ($H \times W \times 3$) image to models for processing. The exact rasterization process can vary between models, often leveraging the RGB channels to represent degree of information. For example [Bansal et al., 2018] represents traffic lights with different shades of gray for each light level. In [Phan-Minh et al., 2020], other agents cars are typically represented in yellow with a fading gradient of yellow representing past positions of each agents.

Other methods have explored various other data representations to aid the trajectory forecasting step. VectorNet [Gao et al., 2020] proposes a new representation for HD-maps which consists in a more semantic vectorized version (see figure 2.1 right). [Liang et al., 2020b] have also opted for exploring different input representations by creating lane graph representations. Lane graph representation uses a graph-based approach to model the road network. This graph explicitly represents lanes and their connectivity, which is more structured and potentially more informative than a rasterized map for understanding complex traffic scenarios. However, explicitly modeling lanes heavily offsets the burden of learning trajectories, as most future trajectories for agents merely follow their current lane except in specific situations.

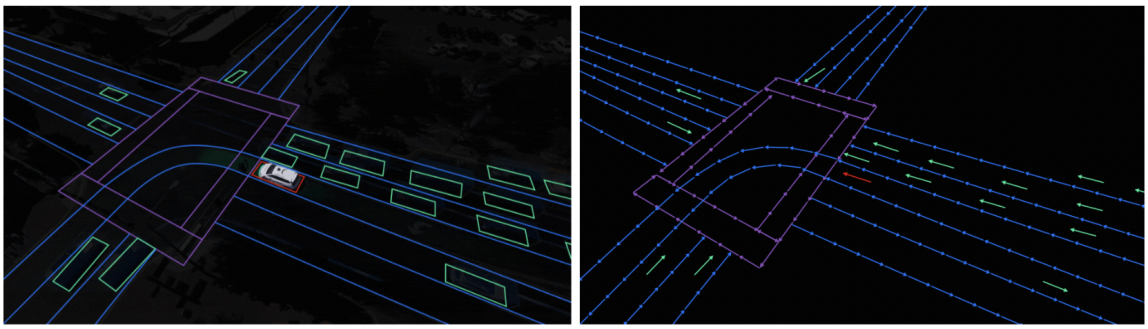


Figure 2.1: **Input data examples**. (left) Rasterized input data, the most prevalent input source, is an RGB image where all semantic elements (road, pedestrian crossings, vehicles, etc.) are fused together in a single image constituting the HD-map. (right) The vectorized version creates a semantically differentiated input data where every element is represented as a series of coordinates.

The problem of different input data is that it two-fold. First, it makes the trajectory prediction task further from the perception task which could make the creation of a powerful end-to-end model more difficult. Second, a rasterized version is a simple RGB image that can be processed by any architecture that can take images as inputs, offering the option to tap into the very wide body of suitable architectures. Any other input data representation needs specialized architecture to be processed, like the vector representation in VectorNet that needs a hierarchical graph neural network to be processed. For these reasons, we chose in this thesis to focus on the more versatile rasterized HD-map input data.

2.1.3 Datasets

Large scale open trajectory forecasting datasets in real world environments are available since 2013 when KITTI [Geiger et al., 2013] was released. As the industry and research progresses, more benchmarks became available, with varying sensors and annotations. Table 2.1 summarizes the characteristics of these datasets.

These datasets are popular benchmarks used primarily for vehicle trajectory forecasting. Other datasets focus on pedestrian trajectories, that are governed by a different set of rules and can be specialized by area. UCY Crowds-by-Example dataset [Lerner et al., 2007], ETH BIWI Walking Pedestrians dataset [Pellegrini et al., 2009], Town Center dataset [Benfold and Reid, 2011], Train Station dataset [Zhou et al., 2012] and Stanford Drone dataset [Robicquet et al., 2016] or TrajNet [Sadeghian et al.,] are all pedestrian-focused datasets. Most of these capture pedestrian movements in world coordinates (as opposed to vehicle coordinates) from drone camera and are used in applications specifically modeling pedestrian behavior, taking into account the modelization of social behaviors that have a greater impact on pedestrian trajectories than vehicle trajectories. Some other datasets focus on specific environments,

Name	Samples	Sensors	Adverse conditions	Annotation
KITTI [Geiger et al., 2013]	15k	Camera (1) 64-b LiDAR	No	Depth Segmentation Detection Road layout
Argoverse [Chang et al., 2019]	44k	Cameras (9) 32-b LiDAR	Yes	Depth Segmentation Detection Road layout Ground height
nuScenes [Caesar et al., 2020]	400k	Cameras (6) 32-b LiDAR RADAR	Yes	Segmentation Detection Road Layout
Waymo Open [Sun et al., 2020]	230k	Cameras (5) 64-b LiDAR	Yes	Segmentation Detection Human skeleton
ONCE [Mao et al., 2021]	1M	Cameras (7) 64-b LiDAR	Yes	Detection

Table 2.1: **Major trajectory forecasting datasets**. Overview of the primary characteristics of popular datasets for trajectory prediction. Adverse conditions: night, rainy or snowy conditions. 32-b and 64-b refer to the number of beams for LiDAR sensors, which affects the point cloud resolution. In this table, the number of samples is one recording of each sensor (e.g. camera images + LiDAR point cloud) along with annotations. To make a trajectory, a number of sequential samples is used; the length of the trajectory can be picked in accordance with comparison imperatives.

for example a surprising amount of datasets focus on roundabouts with openDD [Breuer et al., 2020], the inD Dataset [Bock et al., 2020], or the Round Dataset [Krajewski et al., 2020], highlighting the complexity of such structures in terms of trajectory prediction.

Restricting to the vehicle-centered trajectory forecasting datasets from table 2.1, the differentiating factors lie in the number of samples, the type of sensors and the provided annotations. Even when the same class of sensors are available between two datasets, the quality can vary and be a determining factor in the choice of benchmark. For instance in nuScenes, the LiDAR used is a 32-beam Velodyne, which gives an average point cloud resolution. In ONCE, a 64-beam Velodyne LiDAR is used, which has a much better resolution. As nuScenes has a sizeable amount of data and a more comprehensive set of annotations, including Road Layout, we chose to focus on this dataset since we do not use the LiDAR data.

2.2 Diversity

2.2.1 Diversity in trajectory forcasting

As the domain of autonomous driving evolves, we have seen the perception problems progressing in difficulty, and then trajectory forecasting taking the same path. The initial emphasis in trajectory forecasting was understandably on basic prediction capabilities, primarily aimed at determining the likely future trajectories of other agents. As trajectory forecasting as a research domain matures, it aims to include the complexities of real-world driving, in order to be effective not only on benchmarks, but also in commercial products. Robustness to the myriad of 'corner cases' that make real world driving is the step between good performance on limited benchmarks and good performance in real life.

From deterministic trajectory forecasting, where the goal is to correctly predict the unique ground truth future trajectory from a given past trajectory, the field has been interested in *probabilistic* trajectory forecasting. The addition of a stochastic component is key in this task in order to represent our lack of knowledge in the forecast. As there is inherent uncertainty in the future trajectory not covered by the deterministic ground truth realization, stochastic prediction aims at modeling this uncertainty in a distribution. Historical methods have provided a way to include uncertainty, such as Kalman filters [Kalman, 1960] for a way to include Gaussian uncertainty or Particle Filtering techniques for capturing more complex uncertainty distributions. However, deep learning based methods, largely because of their superior ability to accurately predict future trajectories from high-dimensional data, have supplanted older methods in trajectory forecasting. The stochasticity given by deep learning models often lacks expressiveness, as it is often identifiable to Gaussian noise around the majority mode.

Diverse trajectory forecasting, as an additional task to regular trajectory forecasting, aims at integrating this uncertainty in the prediction in a more explicit way, in order to effectively model possible future scenarios that might differ greatly from the majority mode, such as rare but highly unusual trajectories that would be relevant to predict for security reasons. Prediction diversity is key to make autonomous driving systems more able to navigate the unpredictability of open environments, specifically in the current pipeline (shown in Figure 1.3) by enhancing the robustness and smoothness of planning algorithms. By acknowledging and preparing for a spectrum of possible future movements, autonomous driving systems can make more informed, adaptive, and safer decisions.

As we have seen in table 2.1, most real world datasets providing useful benchmarks for autonomous driving evaluation only have one ground truth future trajectory. With the notable exception of the pedestrian dataset [Liang et al., 2020a], the real-world nature of these datasets make it impossible to capture a distribution of possible futures for one given situation, as even if the acquisition car passes

at the same spot it has before, the other agents have necessarily changed, creating a similar but not identical past conditioning.

The downstream task of trajectory forecasting is planning. Planners, when evaluated in closed-loop [1], are often surpassed by simpler rule-based planners [Dauner et al., 2023]. As such, planners might be one of the weakest link so far in the autonomous chain, although it is changing with the active development of new planners like nuPlan [H. Caesar, 2021] and their adoption by the broader community. As planners are the downstream task from trajectory prediction, we can wonder whether diversity can positively impact them. An interesting work by [Cui et al., 2021] investigated the matter. Like most trajectory forecasting methods, a generative model is used, but the implementation of diversity is interesting in the sense that it is integrated with the planning task. They predict a diverse set of possible future scenarios, and estimate the self-driving vehicle's trajectory by optimizing contingency plans over these scenarios. The planner focuses on comfortable, non-conservative trajectories that ensure safe reactions across various scenarios. The model demonstrates that more diverse motion forecasting result in and safer, less conservative motion plans in evaluations and long-term closed-loop simulations, which highlight the impact that diversity can have over the final self-driving vehicle motion.

In this context, diversity for trajectory forecasting is a useful task that merits some attention. Some works have considered explicitly improving diversity in trajectory forecasting for autonomous driving.

DiversityGAN [Huang et al., 2020a] explicitly targets the diversity of the generated trajectories by curating the latent space of the generator. The generator, comprised of an encoder and a decoder, has a latent space that is shaped using semantic annotations, dividing the space in semantic elements that can then be more easily sampled.

Aside from semantic annotations, other methods have tried to include direct diversity supervision in the output space: [Mangalam et al., 2020] explicitly condition the prediction network on trajectory endpoints. It is done by two components: first a network infers the probable endpoints of a trajectory given its past, then a second network gives endpoint-conditioned trajectory prediction. However the focus on this study was on pedestrian trajectories to also explore the social aspect and it was evaluated on pedestrian datasets. Evaluated on a vehicle-based dataset, [Rhinehart et al., 2018] also conditions the diversity but instead of providing a direct incentive for the trajectories, they try to match the output distribution of predicted trajectories with distribution from the training data, via a Cross-Entropy loss.

Divide-and-Conquer [Narayanan et al., 2021] specifically optimizes for diversity by using external

[1] in which outputs from the previous step are fed as inputs for the subsequent steps, as opposed to open-loop evaluation where the ground truth trajectories are used at each step

information: the lane information contained in many datasets like nuScenes. This ground truth information matches the layout of the scene by providing lane information and how they separate at crossings. It is a very good prior for trajectory forecasting in general and diversity in particular, as how the lane divides in the dataset is a good approximation of what maneuvers a car is allowed to make. However, just as lane information tends to not be used in trajectory prediction due to the fact that it's too strong of a predictor that might not be fully available in real life vehicles that estimate the HD-map, we chose not to use it for diversity either.

Overall, the specialized literature on diversity for trajectory forecasting is scarce, but highlights two major difficulties that we will have to address by tackling this problem. First, diversity is hard to evaluate because there is no ground truth trajectory distribution. To mitigate this problem, several methods have proposed different metrics, based either on self distance between the predicted trajectories or by referring to the coverage of the underlying drivable layout, like the Drivable Area Occupancy (DAO) metric [Park et al., 2020]. Second, the incentive for diversity has to be crafted from something: some methods base themselves on specific objectives using information outside of the past trajectory, like lane or predicted endpoint, some try to arrange preemptively the latent space. In the work presented in this thesis, we try to improve on existing methods by using a more intrinsically motivated diversity mechanism and adapt it to the task at hand.

2.2.2 Architectures for diverse trajectory generation

As the subject of this thesis is not to make an extensive review of existing architectures for generative models in general, we use this section to provide an overview of what is used in the context of trajectory prediction.

Normalizing flows [Rezende and Mohamed, 2015a] have also been used for trajectory forecasting [Ma et al., 2021, Rhinehart et al., 2018, Rhinehart et al., 2019], especially to leverage the exact likelihood computation feature of such models to provide precise conditional predictions or easy modeling of the uncertainty associated with each prediction.

Generative Adversarial Networks (GANs) [Goodfellow et al., 2020] have been used in the context of trajectory forecasting, either pedestrian or vehicle, for a long time [Alahi et al., 2016, Sadeghian et al., 2019, Kosaraju et al., 2019, Huang et al., 2020b].

Variational auto encoders (VAEs) [Kingma and Welling, 2014] are also a popular base architecture for trajectory forecasting. Their versatile nature and extensive body of literature make them suitable architecture for trajectory forecasting: encoders and decoders can be replaced by sequential models like LTSMs to introduce a very efficient inductive bias to produce trajectories. They have been used for trajectory forecasting with much success so far [Lee et al., 2017,

Yuan and Kitani, 2020a, Yuan and Kitani, 2020b, Ivanovic and Pavone, 2019, Salzmann et al., 2020, Tang and Salakhutdinov, 2019, Weng et al., 2020b]. This success can come from the popularity of the underlying method, the ease of use where components can be replaced with different architectures while retaining the same training pattern, and the systematic way of modeling the latent space they exhibit, that allows for exploration of said latent space and potential for manipulating it to better suit the task's need.

Of course, transformers have also been used in the context of trajectory forecasting. While [Giuliari et al., 2021] focuses on pedestrian trajectories, as transformers are more obviously useful to represent social interactions which have more influence on pedestrian trajectories, AgentFormer [Yuan et al., 2021] explores both pedestrian and vehicle trajectories on nuScenes. AgentFormer is one of the first works to leverage transformers to make multi-agent trajectory prediction, and they leverage the transformer architecture to attend to features from past trajectories of any agent, which provides a good way to include interactions between agents.

As our diversity and discovery focus aims to find as many different future trajectories as possible, we didn't need the social component as much, so the majority of this thesis work has a simpler encoder-decoder backbone that is akin to a cVAE. It allows for fast and easy development and a good body of literature to understand the workings of the latent space that would help us analysing it to find useful ways to improve diversity and discovery.

2.2.3 (Conditional) Variational Autoencoders

Much of the work in this thesis delves into enhancing trajectory diversity from generative models, beginning with an exploration of latent spaces within the framework of Variational Autoencoders (VAEs). To facilitate understanding of the content, we provide a quick primer on VAEs in this section.

Variational Inference As the name suggests, VAEs are a class of autoencoders designed for generating new samples that could plausibly come from an unknown real data distribution $p(x)$. In order to do this, we want to model a lower dimensional latent space $\mathcal{Z}$ that represents the underlying features of the data. Having this posterior distribution $p(z|x)$ allows to make predictions about unseen data, generate an accurate predictive distribution and measures of uncertainty for these predictions. However, $p(z|x)$ isn't directly computable, as through Bayes rule it is expressed as $p(z|x) = \frac{p(x|z)p(z)}{p(x)}$, with $p(x|z)$ being the likelihood of seeing x given the latent variable z, $p(z)$ being the prior distribution representing initial belief about how the latent space is organized, usually selected to be a Gaussian prior for ease of computation. Finally, $p(x)$ represents the marginal likelihood of the observed data x over all the latent variables. It is computed through the integration over all possible z values, $p(x) = \int_{\mathcal{Z}} p(x|z)p(z)dz$, not computationally tractable, which is why variational inference is used. Variational inference refers

to the use of a distribution $q(z|x)$ to approximate the true posterior $p(z|x)$. Let's see how VAEs manage to approximate $p(z|x)$.

Variational Autoencoders Autoencoders encode inputs to a lower dimensional latent space $\mathcal{Z}$ and decoding them back to the original input space, promoting feature learning via dimensionality reduction. The Variational part extends autoencoders by adding a probabilistic twist, allowing for the generation of new data points by sampling in the latent space. Given a set of observations x, VAEs utilize a known prior latent distribution $p(z)$ to produce latent codes $z \in \mathcal{Z}$. The objective of VAEs is to approximate the true intractable posterior data distribution $p(z|x)$ with a variational approximation $q_\phi(z|x)$ where ϕ are the parameters of the encoder network.

In order to effectively learn this distribution, VAEs are structured around an encoder and a decoder network. The encoder network $q_\phi(z|x)$ takes an input x and maps it to a distribution over the latent space, represented by z. Effectively, q_ϕ outputs parameters mean μ and variance σ^2 that are used to sample from a probability distribution q that approximates the true posterior distribution $p(z|x)$. The decoder network, $p_\theta(x|z)$ in turn take a latent variable z and reconstructs the input x. The decoder defines the likelihood $p_\theta(x|z)$ of observing x given z, to generate data that resembles the original input as much as possible. By optimizing the parameters θ of the decoder to maximize the likelihood $p_\theta(x|z)$ and the parameters ϕ of the encoder to make $q_\phi(z|x)$ a good approximation to $p(z|x)$, VAEs effectively learn a generative model of the data.

VAEs are trained by maximizing the Evidence Lower BOund (ELBO):

$$\log p(x) \geq \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] + \mathrm{KL}(q_\phi(z|x)||p(z)), \tag{2.1}$$

which is a lower bound on the log-likelihood of the data $\log p(x)$. Maximizing the ELBO with the parameters we can control, $q_\phi(z|x)$ and $p_\theta(x|z)$, indirectly maximizes $\log p(x)$ in a tractable fashion. The ELBO is composed of two main terms, that we find in every VAE-based method:

- the first term $\mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)]$ is the reconstruction objective. It measures how well the decoder network $p_\theta(x|z)$ matches the original input x given the latent code z. A higher log likelihood $\log p_\theta(x|z)$ means the model assigns a higher probability to the actual input x given latent variable z, implying a better reconstruction.

- the second term $\mathrm{KL}(q_\phi(z|x))$ is the Kullback-Leibler divergence between the variational approximation of the posterior distribution $q_\phi(z|x)$ and the prior distribution over latent variables $p(z)$, typically chosen to be Gaussian. This term acts as a regularizer, ensuring both the completeness of the latent space, meaning that sampling any point from the latent space (under the prior

$p(z)$) is likely to result in plausible outputs when decoded; and its smoothness, meaning small changes in latent codes result in small changes in the reconstructions.

This base model is illustrated in figure 2.2 (a), and we will use this type of modelling throughout this thesis to best illustrate the interaction between the different distributions.

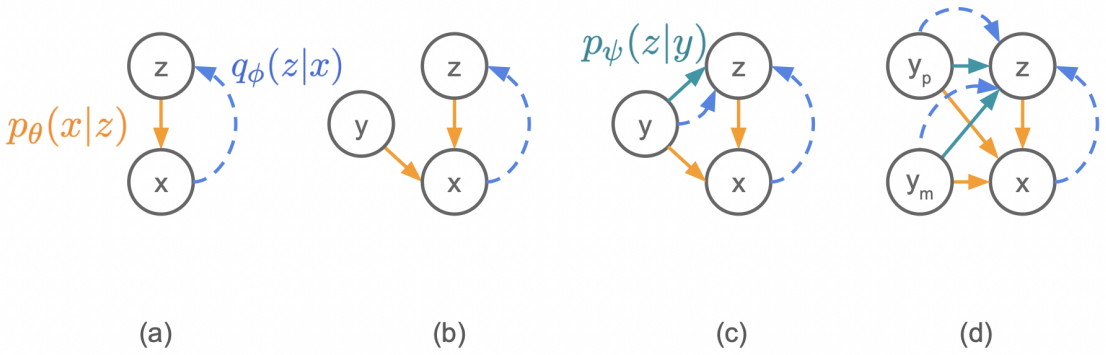


Figure 2.2: **VAE architectures**. Modeling of the interactions between data x, latent code z and conditioning information y. (a) refers to the regular VAE modeling of the data distribution $p_\theta(x) = \sum_z p_\theta(x|z)p(z)$. (b) models the addition of conditioning information y, for cVAEs where we want to learn the conditional posterior distribution $p_\theta(x|y) = \sum_{z,y} p_\theta(x|z,y)p(z)$. (c) shows that the prior $p(z)$ can also be learned conditionally, giving $p_\theta(x|y) = \sum_{y,z} p_\theta(x|z,y)p_\psi(z|y)$ and (d) illustrates the inclusion of distinct conditioning information $p_\theta(x|y_m, y_p) = \sum_{y_m,y_p,z} p_\theta(x|z,y_m,y_p)p_\psi(z|y_m,y_p)$.

Conditional Variational Autoencoders In some cases, we want the generated sample to be conditioned on some y. This conditioning variable could represent any kind of auxiliary information that must be taken into account when generating new data. For instance in the context of trajectory prediction, the autoencoder framework isn't very appropriate and we want x to be the future trajectory, conditioned on the past trajectory y. We then want to approximate the conditional distribution $p(x|y)$, in order to generate data x conditioned on specific information y, as illustrated in figure 2.2 (b, c). The objective function for conditional VAEs (cVAEs) still aims to maximize the ELBO, but does so for the conditional distribution $p(x|y)$:

$$ELBO = \mathbb{E}_{z\sim q_\phi(z|x,y)}[\log p_\theta(x|z,y)] + \mathrm{KL}(q_\phi(z|x,y)||p(z|y)). \tag{2.2}$$

Both terms are similar to the original VAE ELBO objective. The reconstruction term encourages the accurate reconstruction of x from both z and conditioning information y, and the KL divergence term now measures the divergence between the conditional variational approximation $q_\phi(z|x,y)$ and a prior $p(z|y)$ that can depend on y, but is usually simplified to $p(z)$ (figure 2.2 (b)).

cVAEs in the context of trajectory prediction Learning a conditional prior $p_\psi(z|y)$ (figure 2.2 (c)) offers several benefits that can be useful especially in the context of trajectory prediction. It allows the model to capture the dependence of the latent space on conditioning variables y making it more flexible by partitioning the latent space between different conditions. When future outcomes are heavily influenced by the condition, like the future trajectory by the past trajectory, it can become interesting to model these dependencies.

Trajectron++ [Salzmann et al., 2020] is a good example of the many ways the ELBO objective can be adapted for trajectory forecasting. In addition to using a learned prior, the future trajectory is conditioned on both the past trajectory and the layout of the scene, as it is useful in this context to distinguish between dynamic and static conditioning. The two different conditions, respectively y_p for past trajectory and y_m for the map conditioning, are integrated into the framework as illustrated in figure 2.2 (d). The ELBO objective in which we can see how both condition influence the latent code z and the output generation x is as follows:

$$ELBO_{T++} = \mathbb{E}_{z\sim q_\phi(z|x,y_m,y_p)}[\log p_\theta(x|z,y_m,y_p)] + \mathrm{KL}(q_\phi(z|x,y_m,y_p)||p_\psi(z|y_m,y_p)) \tag{2.3}$$

In Trajectron++, one distinctive feature is the use of conditional information y_m and y_p both in the posterior and prior distributions. The posterior distribution $q(z|x,y_m,y_p)$ represents the latent variable distribution given the observed future trajectory x, the layout condition y_m and the past trajectory condition y_p. The prior distribution $p(z|y_m,y_p)$ reflects the model's assumptions about the latent space before observing the future trajectory x, but it is useful to have this prior information conditioned on the past trajectory y_p and layout y_m, in order to avoid using z as merely a Gaussian noise added on the generated x, like when using $p(z)$ as an unconditional prior.

Diversity in cVAEs The conditioning information can potentially overwhelm the latent space, leading to a scenario where the model over-relies on the conditioning inputs and under-utilizes the diversification power of the latent space z. This over-reliance on y_m and especially y_p can result in a lack of diversity in the generated trajectories, as the model relies heavily on the known conditions rather than exploring the range of plausible trajectories that could emerge from a given set of conditions. To counteract this issue, Trajectron++ incorporates a mutual information maximization term in the loss function, between the conditioning information y_m, y_p and the latent distribution z. This term encourages the model to maintain a balance between the influence of the conditioning information and the stochastic nature of the latent space. By doing so, the model not only respects the conditioning constraints but also retains a degree of unpredictability and variability in the generated trajectories.

2.3 Evaluation of diversity

Real-world driving datasets such as nuScenes cannot provide ground-truth distributions of possible futures, as by definition trajectories follow the path of the data acquisition car and we can't have multiple futures for the same past. Even if the car would make multiple passes on the same static road layout, dynamic elements (other cars, pedestrians etc.) would differ. This poses a challenge for evaluating the diversity of proposed trajectories in real-world driving datasets, as we can't compare the generated distribution to a ground truth distribution. Nevertheless, tools have been developed to measure the diversity of the generated trajectories given a single ground truth, while ensuring that the single ground-truth future is among predictions.
After reviewing the literature, we concluded that no single metric could adequately cover "diversity" as a concept and we implement several metrics that all capture a different aspect: intrinsic diversity, diversity with respect to the ground truth, and the admissibility of the different proposed trajectory, a key insight to avoid falling in the trap of ever-expanding diversity at the expanse of meaning.

Ground truth accuracy metrics. First of all, before diversity, we systematically use the min Average and Final Displacement Error (mADE and mFDE respectively) metrics to evaluate the accuracy of the best predicted trajectory, as is customary in the trajectory forecasting literature [Park et al., 2020, Phan-Minh et al., 2020, Yuan and Kitani, 2020a, Yuan and Kitani, 2020b, Bansal et al., 2018, Rhinehart et al., 2 independent from the focus on diversity. mADE is defined as the average of the Euclidean distance between each point of the closest trajectory and the ground truth trajectory, while mFDE measures only this distance between the final point of each.

Diversity and admissibility metrics. Measuring diversity is not straightforward since it is not a well-defined concept. As a consequence, several diversity metrics have been proposed. The ratio of average Final Distance Error (FDE) to min FDE, rF = $\frac{\text{avgFDE}}{\text{mFDE}}$ [Park et al., 2020], is a measure of the spread of the proposed trajectories relative to the ground truth: A high value indicates a high avgFDE, meaning some predictions are far away from the ground truth, and a small mFDE, i.e, one of these predictions is close to the ground truth.

To measure the spread of the predicted set independently from the ground truth, the Average Self Distance (ASD) and Final Self Distance (FSD), introduced in [Yuan and Kitani, 2020a], can be used. Instead of computing the diversity through the constraint of the ground truth, ASD and FSD measure the intrinsic spread of the generated trajectory distribution by calculating the average pairwise distance between all unique pairs of predicted trajectories:

$$ASD = \frac{2}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d(T_i, T_j), \tag{2.4}$$

where N is the total number of predicted trajectories and $d(T_i, T_j)$ denotes the distance between trajectories i and j. FSD is computed following the same principle, only taking the distance between final points of each trajectory.

A qualitative assessment of the diversity can also be made by two additional metrics: the Drivable Area Occupancy (DAO) [Park et al., 2020], which measures the diversity in predictions that are in the drivable area and the Drivable Area Count (DAC) [Chang et al., 2019] defined as DAC $= \frac{N-m}{N}$, where m is the number of predictions that exit the drivable area (DA).

The DAC metric is related to the $OffRoadRate$ sometimes seen in the literature [Narayanan et al., 2021, Greer et al., 2021, Deo et al., 2022, Naumann et al., 2023] to refer to the same quality. The Offroad Rate computes the percentage of trajectories that fall outside of the DA and is thus interchangeable with the Drivable Area Count with the following formula: OffRoadRate $= 1 - DAC$.

As discussed in [Park et al., 2020], these metrics can be seen as providing complementary information about diversity: rF, ASD and FSD quantify the diversity in terms of spread; DAC only assesses the admissibility of the trajectories in the set; DAO captures a mix of diversity and admissibility, by measuring the spread among admissible trajectories only. DAO and DAC, being directly related to the drivable area, they provide valuable insights into how the proposed trajectories resemble real trajectories.

2.4 Discovery

The diversity task is already an established goal in trajectory prediction and other domains. It essentially aims to create methods that find all data modes in the training data that match the conditioning, in order to produce an output distribution of generated trajectories that is as faithful as possible to the range of possible elements in the training set. In the context of trajectory forecasting, these modes could be the "type" of trajectory: going left, right, straight, accelerating, decelerating, etc. Even if for any given situation there is a majority mode that is more likely (i.e. trajectories are heavily biased towards straight trajectories), diversity objectives aim at producing trajectories that also represent minority modes. Modeling the posterior distribution with smaller rare but dangerous modes can be an effective way of improving the reliability and security of autonomous driving models, and discovery can help provide a way to discover those modes relying on external principles and not training dataset where rare modes (especially dangerous situations) are often not represented.

However, one can ask the question: **what if a minority mode isn't represented in the training data?** In this case, no diversity method can include such an unknown trajectory in the generated future distribution, as it has not been seen before. The concept of discovery is a more exploratory task that aims at generating elements that are not in the training distribution, but still relevant in some way.

As it is related to extrapolation and therefore a difficult problem, the topic of discovery hasn't been studied yet in the context of trajectory prediction. However, several works have hinted towards an analysis of existing models on simpler tasks on subjects that can be considered as being discovery-related. The present section aims at providing an overview of the body of literature on this subject, and drawing bridges to the current task.

2.4.1 Out-of-distribution (OOD)

As the trajectories we're interested in generating are clearly out of the training distribution, we can turn to the out-of-distribution (OOD) literature to understand what has been done on the subject of trajectory forecasting. There are very little works on the subject adapted to trajectory forecasting but the recent work [Wiederer et al., 2023] studies out-of-distribution detection in this context. In this work, the diving scene is encoded and used to predict a conditional distribution of trajectories, that can be then used in comparison with the actual generated trajectories to detect whether OOD trajectories have been generated, given the distribution shift. It uses the Shifts [Malinin et al., 2021] dataset, which has been specifically designed to evaluate the robustness in trajectory prediction given shifting distribution like adverse weather conditions.

[Wiederer et al., 2023] presents a method that combines trajectory prediction with OOD detection and uncertainty estimation, particularly focusing on automated driving scenarios. The method uses a two-phase training process and a scene encoder to predict the conditional distribution of trajectories, with additional modules for OOD detection and uncertainty estimation. This approach aims to improve the reliability of trajectory predictions, especially in complex traffic scenes. The method was evaluated using the Shifts dataset, which is unique for OOD detection in trajectory prediction, and includes data affected by distribution shifts like adverse weather conditions.

However, OOD literature mostly focuses on OOD *detection*, whereas we're interested in OOD *generation*. Most methods that are used in detection, especially since most are in the context of image OOD detection, could not be applied to the problem of discovery especially for trajectory forecasting. Nevertheless, the parallels in the different tasks still exist and methods from OOD detection could be used in the context of discovery to identify shifting trajectories and integrate them back in the training distribution if they are interesting enough. This avenue wasn't explored in this thesis but it could be an interesting area of research to bridge both tasks with common methods.

2.4.2 Combinatorial generalization

Even if the discovery of trajectories unseen in the training dataset have not been explored a lot in the literature, there exist some works regarding the analysis of generative models and especially the

generalization to unseen combinations of features in the dataset.

Combinatorial generalization refers to the ability of a neural network to recombine learned elements in novel ways to produce combinations that have not been seen during training. Usually given any task, a model that has good combinatorial generalization correctly represent the underlying patterns and relationships in the training data rather than memorizing it. This modelization then help make a model more versatile and robust to real-world combinations. A model exhibiting a high combinatorial generalization capability can be more data efficient because not every possible combination has to be present in the dataset, which is a characteristic that we want in discovery.

[Montero et al., 2020] is an interesting paper, delving into the details of combinatorial generalization and trying to formalize the different tasks and how models specifically designed to handle combinatorial generalization, like β-VAE [Higgins et al., 2017] fare in this regard. Figure 2.3, taken from the paper, highlights their classification of three different levels of generalization, in increasing degree of complexity.

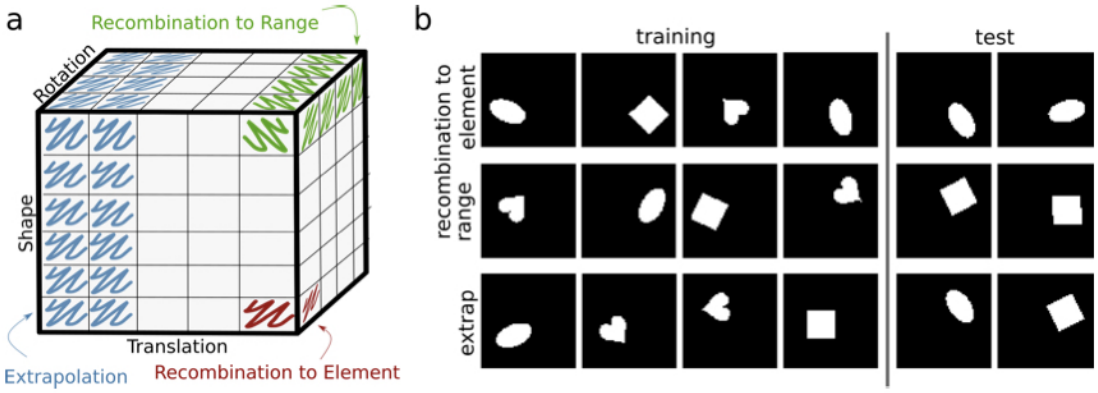


Figure 2.3: **Combinatorial generalization tasks classficiation**. [Montero et al., 2020] task classification for the simplified case where 3 generative factors are involved in the generation of new examples. (a) 3D view of each combination on a cube to visualise combinations removed by the different tasks defined, in increasing complexity order: Recombination-to-element, Recombination-to-range, Extrapolation. (b) View of

Given an idealized $\mathbb{R}^N$ space where each axis is a generative factor ($N = 3$ in figure 2.3) for the element (image) we want to generate, each combination of N factors represent a combination we want to be able to generate. In the dSprites dataset [Matthey et al., 2017] that is specifically designed to test for the disentangling of each generative factor in the context of images, one datapoint can be a combination of shape (heart, square, ellipse), position in both x and y axis on the image, rotation, and scale. The first task is called "recombination-to-element", and is the easiest of all. It involves removing from the training dataset a complete N-uple and trying to generate it. For instance, the combination "ellipse / bottom-left / 0° rotation / scale 1" has not been seen in the training data and

we want to generate it. Since other similar combinations like "square / bottom-left / 0° rotation / scale 1" or "ellipse / bottom-left / 30° rotation / scale 1" have been seen, it is the easiest generalization task and the paper indeed shows that most models, regardless of disentanglement capability, manages to generate the missing combination. The second task, "recombination-to-range", is more difficult as it involves removing a whole range of examples, like removing all small-sized ellipses regardless of their position or rotation. For this task, most methods fail, even if they are specifically designed for disentangling and have a good overall disentangling score measured by the metric of factor separation accuracy. Most of the paper studies this intermediate task, which still qualifies as interpolation because the model is required to fill in gaps within the observed range of the data, rather than extrapolating beyond it. Mentioned in the classification but not studied, the last task of Extrapolation refers to the removal of a broader range of values across multiple modalities (axis) in the training dataset: for instance, all shapes left of the image are removed and we want to be able to reconstruct any image that has an object on the left, a task that all studied models fail.

The followup work to this initial study, [Montero et al., 2022], examined the source of such combinatorial errors, to assess whether the encoder or the decoder was more at fault. The outcome of this study was to characterize both types of errors but didn't give a definitive answer as to whether the encoder and decoder was more responsible: the errors were often very intertwined despite the separable nature of the generative factors of the image datasets and highlighted the difficulty of the combinatorial generalization task, even when it didn't involve extrapolation.

2.4.3 Extrapolation

The definitions of [Montero et al., 2020], illustrated in figure 2.3, provide a good starting point to think about extrapolation. As discovery is extrapolation, we focus more on this side of the problem. However, [Montero et al., 2020] and [Montero et al., 2022] were more focused on the recombination-to-element and recombination-to-range tasks, illustrating that most methods failed the latter, and didn't focus on the extrapolation part. We would like to expand their definitions in the Extrapolation task by distinguishing between to very different tasks that under their framework could both be labeled as Extrapolation. Both tasks are illustrated in figure 2.4. As extrapolation is defined as removing an entire range of values from all the combinations, one could argue that removing the "middle" values of a modality (right panel of figure 2.4) counts as extrapolation. We would like to emphasize that this is not the case, as removing middle values is more akin to interpolation than extrapolation, the latter implying that the range of values we try to discover is *outside* the range of values in the training dataset.

It has been shown that it's not possible to learn a fully disentangled latent representation without the addition of bias [Locatello et al., 2019], but the study in [Montero et al., 2022] showed that even

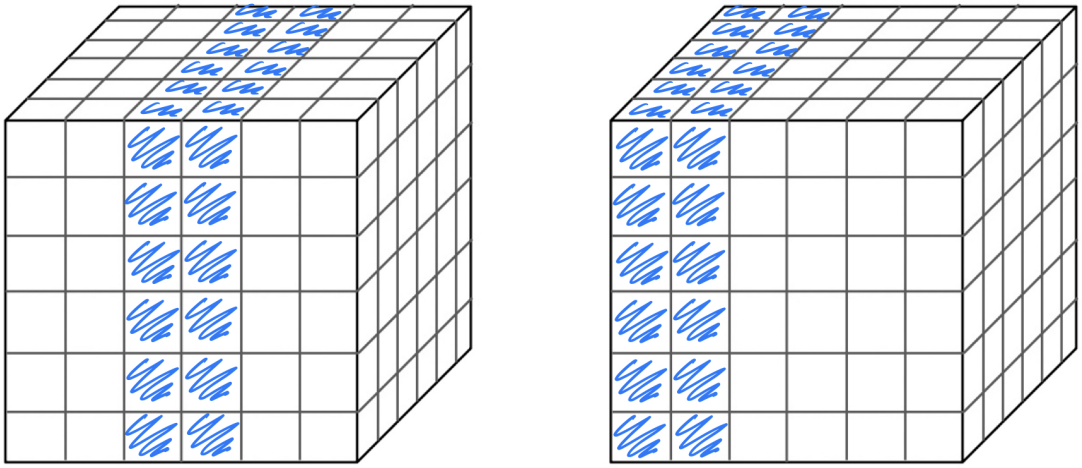


Figure 2.4: **Extrapolation slicing**. To further the definition of what constitutes Extrapolation, we point that the number of elements missing from the training dataset doesn't constitute the full picture: "where" these elements are removed is also a crucial element: removing middle elements that can be interpolated from the remaining combinations (left) doesn't characterize an extrapolation task, whereas removing all elements from one value onwards does (right).

a fully disentangled ideal model failed to perform combinatorial generalization (and by extension extrapolation) convincingly, highlighting the fact that perfect disentangling and separation of generative factors wasn't the primary challenge in the quest for models that are able to extrapolate. Given these classifications and insights, we delve into the discovery exploratory task with caution in chapter 4, not by the disentangling side, but by the external conditioning side.

2.5 Conclusion

In summary, diversity and discovery, the main topics of this thesis, are two facets of the same line of improvement for autonomous driving systems: modeling rare but dangerous minority modes can provide an important tool for making planners less conservative and more human-like, by reducing uncertainty [Cui et al., 2021]. The first step is to improve the generative distribution diversity by accurately represent training diversity, which is done in the trajectory forecasting literature by adding a diversity focused component to an existing trajectory forecasting generative model [Park et al., 2020, Yuan and Kitani, 2020a]. Discovery is a more exploratory task, which can be tackled under the prism of works outside of the trajectory forecasting literature like [Montero et al., 2020], aiming to clarify the theoretical workings of latent space formation in VAE-based models.

Chapter 3

Diversity in generative models

Chapter Abstract

Predicting multiple trajectories for road users is important for automated driving systems: ego-vehicle motion planning indeed requires a clear view of the possible motions of the surrounding agents. However, the generative models used for multiple-trajectory forecasting suffer from a lack of diversity in their proposals. To avoid this form of collapse, we propose a novel method for structured prediction of diverse trajectories. To this end, we complement an underlying pretrained generative model with a diversity component, based on a determinantal point process (DPP). We balance and structure this diversity with the inclusion of knowledge-based quality constraints, independent from the underlying generative model. We combine these two novel components with a gating operation, ensuring that the predictions are both diverse and within the drivable area. We demonstrate on the nuScenes driving dataset the relevance of our compound approach, which yields significant improvements in the diversity and the quality of the generated trajectories.

The work described in this chapter gave rise to the following publication:

Laura Calem, Hedi Ben-Younes, Patrick Pérez, Nicolas Thome. "Diverse Probabilistic Trajectory Forecasting with Admissibility Constraints". In International Conference on Pattern Recognition (ICPR), 2022.

Contents

3.1 Introduction

3.1.1 Context

In trajectory forecasting for autonomous vehicles, future prediction is inherently stochastic since the human or automated driver has only access to very partial information about other road users' intents. It is also often multi-modal, since several admissible, yet very different driving actions can be taken at any instant by each agent. This is especially true at intersections where there are multiple directions available in the drivable area, but it can also be true on an unidirectional road, for example if the layout widens, if it has several lanes in which the considered vehicle can go, or simply if the speed of the traffic varies.

Intuitively, ignoring part of these possible future trajectories can hinder an autonomous or assisted driving system. The main downstream task of trajectory prediction is planning, that gives the final steering and acceleration controls to the ego-car. It has been shown in [Cui et al., 2021] that, when provided with a desired direction, the planning improves significantly by incorporating the varied predictions of other vehicles' trajectories compared to a deterministic approach where only the most likely trajectories are provided. The diversity isn't used for predicting diverse ego-car trajectories, but for all other agents in the scene to assess how each possibility impacts planning, improving said planning of the ego-car. Diversity in planning mitigates several issues arising in planning for

autonomous vehicles such as conservative driving and braking smoothness. To improve planning but also the accuracy of trajectory forecasting, multiple-output forecasting models have emerged. Accuracy-focused multiple-output models don't exhibit much diversity due to them not being trained with that objective in mind, as datasets only typically have only one grand truth and not a distribution of ground truths for a single past trajectory. While having good ground truth accuracy is of course an important milestone for autonomous vehicles, the diversity of the proposed set can lead to a better assessment of the driving scene and thus to a better and smoother planning. Corner-case trajectories, like u-turns that rarely happen but are legal, or vehicles turning on smaller roads or driveways, aren't a decisive factor in overall accuracy since they're so rare, but predict them can add robustness to the subsequent planning [Cui et al., 2021]. Thus, the focus of this thesis is on the diversity part of the multiple-output forecasting, in order to predict a limited number of future trajectories that capture well the available driving options for the near future. A crucial aspect for both safety and corner-case accuracy is thus to control the diversity of the proposed trajectory set.

For security and corner-case accuracy, semantic diversity appears critical, and it is this diversity task that this thesis and this first work aims to tackle.

3.1.2 Diverse set generation

The shift towards stochastic future trajectory prediction from deterministic future trajectory prediction started around 2016 with the use of various architectures that can be labeled as generative autoencoders, who can sample multiple future trajectories [Lee et al., 2017, Park et al., 2020, Salzmann et al., 2020]. However, the output distribution that such models provide sticks by construction to the one in the training data, which is mostly unimodal if real driving recordings are used: only a single future exists for a given past trajectory. At a higher level, some types of trajectories, such as turning rather than driving straight at an intersection, are severely under-represented. However, in many settings such as crossings, different driving actions can be taken, leading to several different yet admissible trajectories.

In trajectory prediction datasets, by essence, most future trajectories are simply a continuation of the past trajectory in a straight line, creating a majority mode that is usually for the vehicle to go straight. In datasets, scenarios on which models are trained are created by cutting a fixed time window from longer driving scenes, leading to repetition and overly straight-biased trajectories, like in nuScenes where the 1000 20-second long scenes are separated in multiple overlapping shorter scenes of 3-seconds past and 6-seconds future horizons. 70% of the resulting scenes exhibit a no-curvature linear trajectory, as measured by computing the curvature of the polyline (more details given in section 5.2.2). Correctly predicting turning scenarios, especially in complex situations like crossings, doesn't always align with the accuracy metrics for predicting future trajectories. As a result, many studies tend to

overlook semantic diversity, often either neglecting it or merely introducing noise in the generative process.

Many generative models used for trajectory prediction [Lee et al., 2017, Chai et al., 2019, Salzmann et al., 2020] *e.g.*, based on generative adversarial networks (GANs) [Goodfellow et al., 2014] or variational auto-encoders (VAEs) [Razavi et al., 2019, Lucas et al., 2019], have no explicit control on the diversity beyond the one of the data distribution. Therefore, the dominant mode will be sampled every time, a phenomenon akin to GAN [Goodfellow et al., 2014] mode collapse regime sometimes described in the context of VAEs as posterior collapse. The problem is exacerbated in the context of real world driving datasets where the overwhelming majority of trajectories are continuation of the past trajectory This observation, illustrated in Figure 3.1 (left panel), motivates our approach for designing a probabilistic model based on a more structured diversity.

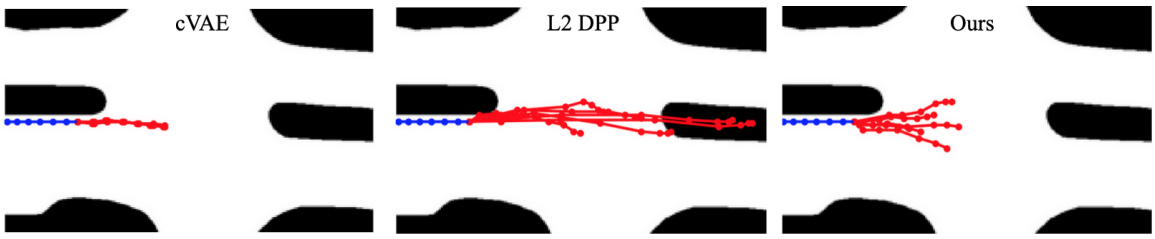


Figure 3.1: **Effect of different methods on diversity**. Given a vehicle's known past trajectory (blue) and the road layout (black and white map), multiple futures are predicted (red). (Left) In real datasets, a single future trajectory is available in training, making a standard generative model such as a conditional VAE (cVAE) unable to sample admissible options far away from the supervision. (Middle) Although Determinantal Point Processes (DPPs) [Kulesza and Taskar, 2012] are appealing for sampling diverse predictions, using a standard ℓ_2 kernel as in [Yuan and Kitani, 2020a] induces mostly *longitudinal* variations and may overshoot in non-drivable areas. (Right) In the proposed method, DIVA, the designed DPP kernel also considers the *lateral* deviation at destination between two trajectories, and explicitly penalizes predictions outside the drivable area when training the diversity model. Consequently, DIVA samples driving options that are diverse, including steering-wise, and admissible.

This is the first key issue that diversity research tries to tackle: going from a deterministic to a stochastic setting "improves" diversity simply by having a distribution of possible futures rather than only one prediction, but this diversity isn't controlled or semantic in any way.

3.1.3 Predicted set consistency

Probabilistic trajectory prediction is a task often based on generative models composed of an encoder, projecting high-dimensional input data (such as bird-eye view maps and past trajectory data) to lower-dimensional embeddings. Then, for future prediction, these are processed by a decoder,

which outputs future trajectories. The stochasticity in the prediction comes from concatenating a sampled latent vector with these embeddings before decoding. In order to make N future predictions, one typically samples N latent codes sequentially from the same distribution, producing N different future trajectories.

As these latent codes are sampled sequentially, there is no consistency between the whole set of predicted future trajectories, hindering the system's capacity to output a meaningful set of future predictions, where ideally each future trajectory would represent a different driving option (e.g. direction or speed). Previous work [Yuan and Kitani, 2020a] explored the use of Determinantal Point Processes (DPPs) [Macchi, 1975] in the context of trajectory prediction.

3.1.4 Proposition

In this first work, we introduce a new method of DIVerse trajectory prediction with Admissibility constraints (DIVA) for probabilistic forecasting of road users. In particular, our approach allows the sampling of the main relevant modes of the future trajectory distribution, as illustrated in Figure 3.1. To achieve this goal, our contributions are:

- We introduce a diversity sampling function (DSF) based on a DPP [Kulesza and Taskar, 2012]. The diversity is explicitly controlled through the definition of the DPP kernel. In particular, we introduce a new kernel adapted to the task at hand, which enforces trajectories' end-points to be far away in the *lateral* direction (amounting to steering diversity) rather than in the *longitudinal* one (amounting to speeding diversity).
- We also control the "quality" of the sampled trajectories via a loss that penalizes violations of the driving area's topology. We learn quality and diversity-based latent codes which we merge with a gating fusion mechanism. This enables the quality loss to filter out irrelevant trajectories predicted outside of the drivable area.
- We evaluate the performance of our system on a real-world dataset (nuScenes [Caesar et al., 2020]) with a broad selection of metrics, demonstrating that trajectories that are both diverse and admissible are well produced.

3.2 Related work

Diversity. A growing body of research [Chai et al., 2019, Zhang et al., 2013, Zhao et al., 2019, Weng et al., 2020a, Robicquet et al., 2016] involves predicting a distribution of future trajectories rather than a univocal future. Many of these methods build upon an encoder-decoder architecture with sampling in the latent space, either with a traditional cVAE [Lee et al., 2017] or with more elaborated techniques [Alahi et al., 2016, Park et al., 2020, Salzmann et al., 2020]. [Ramasinghe et al., 2021] provide

a mechanism for modeling the latent space as a continuous multimodal space, but assume that a distribution of admissible ground truths for each training example is available. This is often not the case in real-world driving datasets.

Several strategies have been applied to overcome this limitation. In MTP [Cui et al., 2019], a multi-output architecture is proposed, trained to encourage each mode to specialize for a distinct behavior. In recent work [Kim et al., 2021], the lane information is used as a prior for semantic behavior decision, thus providing feasible and diverse trajectory forecasts. Park *et al.*[Park et al., 2020] use a normalizing flow [Rezende and Mohamed, 2015b] decoder, and approximate the true distribution of future trajectories using the whole drivable area instead of the single ground truth, which encourages sample diversity. CoverNet [Phan-Minh et al., 2020] tackles the issue of diversity by predicting trajectories as distinct classes, where the set of possible categories is chosen to maximize the coverage on a training set. Another line of approaches uses DPPs to increase the diversity in the set of predicted trajectories. DPPs, introduced in [Macchi, 1975] in the context of particle physics, are probabilistic models which recently gained the attention of the machine learning community [Kulesza and Taskar, 2012, Mariet et al., 2019, Robinson et al., 2019, Celis et al., 2016]. They have been explored for various applications such as video subset selection [Gong et al., 2014], document summarization [Hong and Nenkova, 2014], or time series forecasting [Guen and Thome, 2020]. GDPP [Elfeki et al., 2019] provides an interesting way to build the DPP kernel by matching the true diversity of the data. However, this method requires access to the ground-truth distribution of the data, which is not available in real-world driving datasets. In the context of trajectory forecasting, DPPs have been used with cVAEs in [Yuan and Kitani, 2020a] and with Graph Neural Networks in [Weng et al., 2021]. In our work, we also use a DPP to improve the diversity of the predicted trajectories. We depart from these previous works by incorporating scene information in the DSF, which guides the sampling towards more admissible regions.

Admissibility. Several works explore using physical constraints to guide trajectory generation. In Neural Motion Planner [Zeng et al., 2019b], candidate trajectories are sampled in the space of *clothoids*, which ensures that they are dynamically feasible. CoverNet [Phan-Minh et al., 2020] generates a set of possible future trajectories by integrating the dynamic state of the vehicle. Park *et al.*[Park et al., 2020] generate physically-admissible trajectories by setting a low acceleration prior on the predictions. While having no explicit control for admissibility, Salzmann *et al.*[Salzmann et al., 2020] constrain the outputs to be admissible under the vehicle's current dynamic state. Our work differs from these works as we define admissibility with layout constraints in addition to dynamic feasibility.

3.3 Problem formulation

Given the T_p past (and current) 2D positions of an agent and a "map" of its current environment, the multi-output forecasting task amounts to predicting N possible trajectories over the T_f future instants. Denoting $\mathbf{S} = (\mathbf{S}_\mathrm{p}, \mathbf{S}_\mathrm{f}) \in \mathbb{R}^{(T_\mathrm{p}+T_\mathrm{f})\times 2}$ the agent's trajectory over the whole time interval and $\mathbf{M} \in \mathbb{R}^{H\times W\times 3}$ the environment map centered on agent's current position $\mathbf{S}_\mathrm{p}(T_\mathrm{p})$ (using an RGB encoding of all static and dynamic elements in the scene, see example in Figure 3.2), the forecasting model is trained on example pairs $(\mathbf{S}, \mathbf{M})$. At runtime, it must predict for each agent in the scene N future trajectory samples, $\hat{\mathbf{S}}_\mathrm{f}^{(n)}, n = 1 \cdots N$, given $(\mathbf{S}_\mathrm{p}, \mathbf{M})$. Following [Park et al., 2020], the temporal horizons in our experiments are set to $T_\mathrm{p} = 12$ and $T_\mathrm{f} = 6$, which amounts to 6 and 3 seconds respectively at 2Hz, and the number of predictions is $N = 12$.

While our method is agnostic to the specific architecture of the underlying generative model, we chose for our experiments a simple conditional variational autoencoder (cVAE), as done in [Lee et al., 2017] for trajectory prediction, which we adapt to suit our specific needs, as explained next.

3.4 DIVA

We detail here the DIVA model for diverse trajectory prediction with admissibility constraints. DIVA builds upon a generative model to construct a latent space from which to sample codes representing future trajectories (section 3.3). We then describe in subsection 3.4.4 the proposed method for introducing a structured diversity via a DPP kernel, while controlling the quality of the forecast with respect to the drivable area.

3.4.1 Encoding

At a given instant and for a given agent in the scene, the encoding block takes $(\mathbf{S}_\mathrm{p}, \mathbf{M})$ as input. The past trajectory is encoded by a gated recurrent unit (GRU) network [Cho et al., 2014], as $\mathbf{h} = \mathrm{GRU}(\mathbf{S}_\mathrm{p})$, where $\mathbf{h} \in \mathbb{R}^{d_h}$ is the last hidden state of the recurrent network. The map of the agent's environment is processed by a convolutional neural network to produce an embedding $\mathbf{m} = \mathrm{CNN}(\mathbf{M})$ used as local physical constraints.

3.4.2 Sampling and decoding

Both embeddings $\mathbf{m}$ and $\mathbf{h}$ are concatenated and used to predict the parameters μ and σ of the Gaussian distribution over latent codes $\mathbf{z} \in \mathbb{R}^{d_z}$. A sampled latent code is then concatenated with $\mathbf{m}$ and $\mathbf{h}$ to produce the initialization for the hidden units of the decoder recurrent network. Finally, the output

of this RNN decoder is passed through a series of fully-connected layers to produce the final trajectory $\hat{\mathbf{S}}_\mathrm{f}$. In effect, N latent codes are sampled for a given $(\mathbf{m}, \mathbf{h})$, yielding N distinct future trajectories.

3.4.3 Training the generative model

To train the underlying generative model, we use the VAE loss introduced in [Kingma and Welling, 2014], adapted to include both inputs $\mathbf{S}_\mathrm{p}$ and $\mathbf{M}$ and to reflect the predictive nature of the task rather than an autoencoding one:

$$L_\mathrm{cvae}(\phi, \theta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{S}_\mathrm{p},\mathbf{M})}[\log p_\theta(\hat{\mathbf{S}}_\mathrm{f}|\mathbf{z}, \mathbf{S}_\mathrm{p}, \mathbf{M})] - \mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{S}_\mathrm{p}, \mathbf{M})\|p(\mathbf{z})), \tag{3.1}$$

where ϕ and θ are the parameters of the encoder and decoder respectively. The first term is the likelihood of the predicted trajectory and can be seen as a reconstruction quality term; the second term is the Kullback-Leibler divergence between the learned latent distribution q_ϕ and a prior $p(\mathbf{z})$, generally chosen to be Gaussian [Higgins et al., 2017, Lee et al., 2017] for ease of sampling from this prior. A generative model alone usually suffers from mode collapse, as no incentive is provided to produce diverse samples. In that case, the trajectories generated by the model concentrate around the main mode from the underlying trajectory distribution, as illustrated in Figure 3.6.

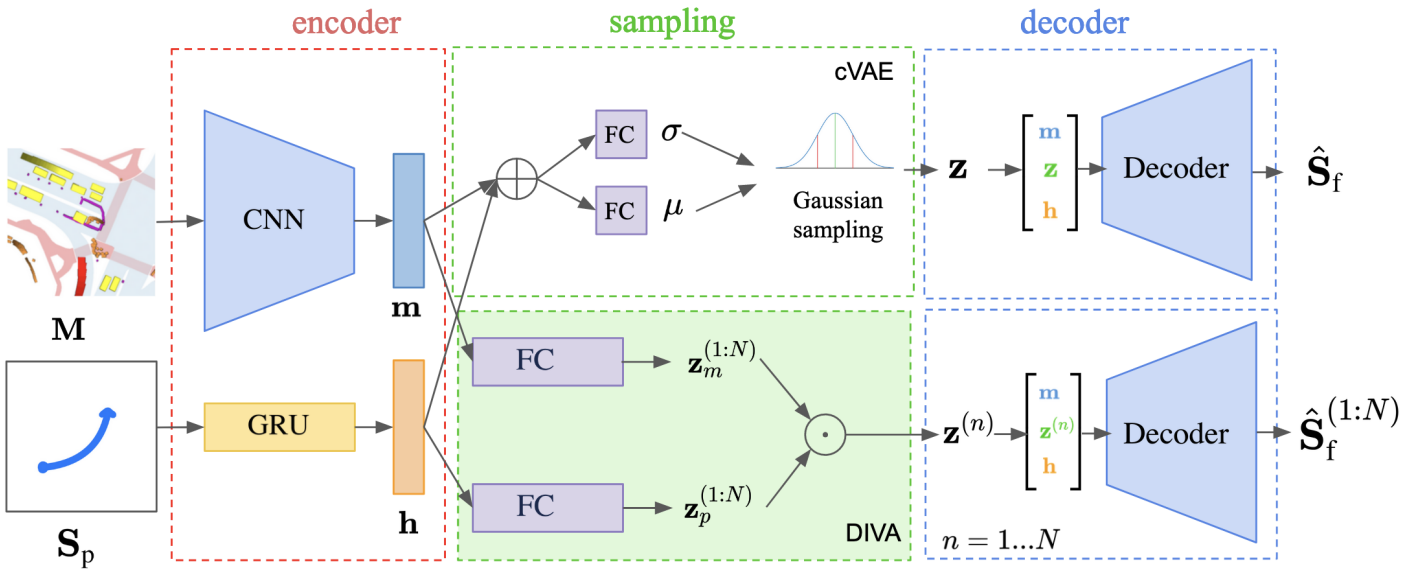


Figure 3.2: **General architecture of the proposed trajectory prediction method in DIVA.** The upper part of the figure describes the underlying generative model, here a cVAE adapted to include layout information $\mathbf{M}$. The lower part of the figure shows the proposed diversity sampling function that replaces the sampling part. $\oplus$ and $\odot$ denote concatenation and element-wise product, respectively.

3.4.4 Structured diversity with physical constraints

Given a trained generative model, we propose to replace the sequential random sampling from the prior $p(\mathbf{z})$ with a diversity sampling function (DSF) trained to predict multiple $\hat{\mathbf{S}}_\mathrm{f}$'s jointly. As illustrated in the lower part of Figure 3.2, the DSF is implemented as a small two-branch feed-forward neural network. In contrast with the generative model sampling, where the N latent codes are sampled independently in $\mathbb{R}^{d_z}$, the network is designed to output all the latent codes at once, producing an output in $\mathbb{R}^{N\times d_z}$.

In order to structure the diversity of the proposed trajectories, we split the DSF between diversity and quality, with each branch controlling a partial latent code. The diversity branch takes the representation of the past trajectory, $\mathbf{h}$, and produces N partial latent codes $\mathbf{z}_p^{(n)}$, whereas the quality branch takes the map representation $\mathbf{m}$ and gives N partial latent codes $\mathbf{z}_m^{(n)}$. The two associated partial codes are then combined using an element-wise product to produce a final latent code $\mathbf{z}^{(n)}$. Through this gating mechanism, the map-specific constraints are imposed to the diverse set of trajectories. The corresponding training loss,

$$\mathcal{L}_\mathrm{dsf} = \lambda\mathcal{L}_\mathrm{dpp} + (1-\lambda)\mathcal{L}_\mathrm{layout}, \tag{3.2}$$

is naturally comprised of two terms. The first term, $\mathcal{L}_\mathrm{dpp}$, favors the diversity through an adapted DPP kernel and the second term, $\mathcal{L}_\mathrm{layout}$, injects the quality constraints structuring this diversity. $\lambda \in (0,1)$ is a parameter controlling the tradeoff between the two losses, as discussed in greater detail in subsection 3.5.4. The following sections describe each loss component in greater detail.

3.4.4.1 Diversity with a DPP kernel

The first term, promoting diversity, relies on Determinantal Point Processes (DPPs), for which we need to provide some background and notation, summarizing from [Kulesza and Taskar, 2012], in order to give enough context for our proposed method.

Determinantal Point Processes (DPPs) are a class of probabilistic models designed for sets. For many applications including ours, the interesting feature of DPPs is their ability to explicitly handle negative correlations among the elements ("points" in the dame Point Process) within these sets, while having efficient sampling algorithms. DPPs were first used in the context of particle physics [Macchi, 1975] for their ability to model the repulsion between particles: contrary to sampling a uniform distribution in space, which results in some level of clumping, sampling a DPP results in a more uniform spread, as illustrated in figure 3.3 (made with the `pyDPP` package [1]).

Formally, given a countable ground set Y of "items", a DPP is a probability measure over the power set $P(Y)$ of Y, where each subset of Y is assigned a probability of being drawn. In order to

[1] https://github.com/satwik77/pyDPP

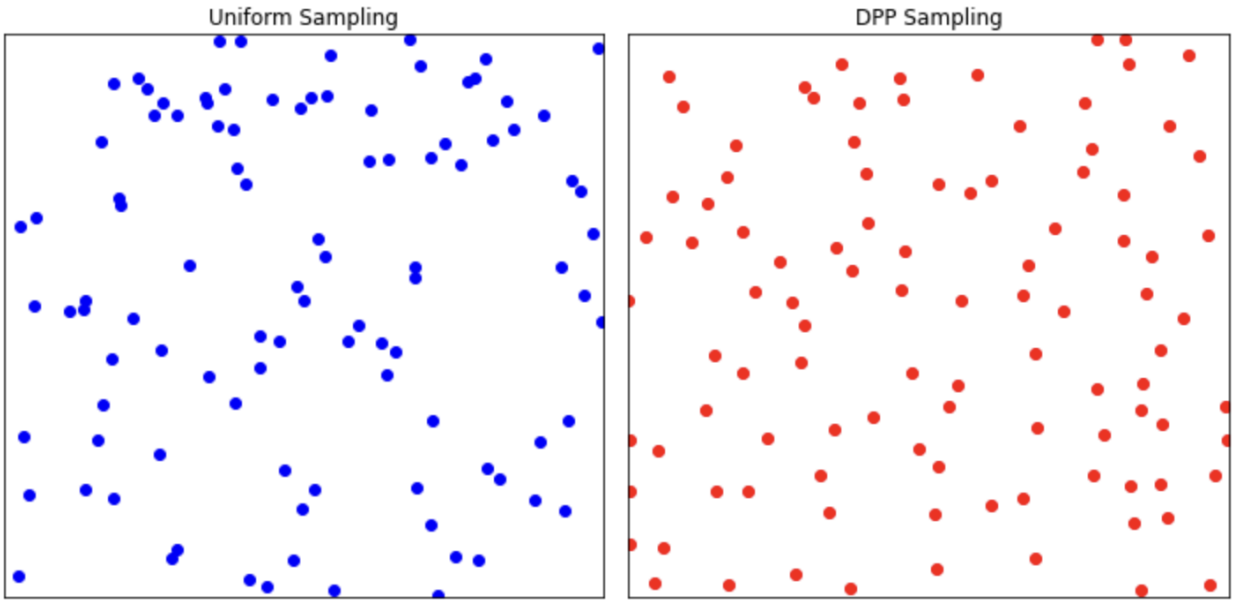


Figure 3.3: **Determinantal Point Processes effect 2D illustration**. Illustration of the repulsive effect of DPPs on 2D points (right) compared to a uniform sampling (left).

use DPPs, we first need a kernel function K, modeling correlations between elements of Y. While $K(i,j)$ for $i \neq j$ gives the correlation or similarity between elements i and j, $K_(i,i)$ gives the marginal inclusion probability of element i.

These correlations can be leveraged in the defining property of DPPs, stating that if $\mathcal{P}$ is a determinantal point process and B a random subset drawn according to $\mathcal{P}$, we have for any part $A \in P(Y)$:

$$\mathbb{P}[A \subseteq B] = \det \mathbf{K}_A, \tag{3.3}$$

Where $\mathbf{K}_A$ is the $|A| \times |A|$ kernel matrix representing the evaluation of K restricted to elements of A.

Through this definition, the repulsive effect of DPPs can be seen easily, by taking the example of a 2×2 subset A,

$$K_A = \begin{bmatrix} P(i) & P(i,j) \\ P(j,i) & P(j) \end{bmatrix}; \tag{3.4}$$

for which the probability of inclusion is given by:

$$\mathbb{P}_K[\{i,j\} \subseteq B] = K(i)K(j) - K(i,j)^2. \tag{3.5}$$

The marginal probabilities of each element increase the overall inclusion of both according to their value, and the correlation between the two (usually symmetric) decreases this probability. Thus, elements which have a low probability result in an overall low probability of simultaneous inclusion, and two high-probability elements also result in an overall low probability if they are too similar. The sampling naturally favors subsets for which both the marginal probabilities are high and similarity low.

The determinants of the submatrices $\mathbf{K}_A$ for all $A \in P(Y)$, also called principal minors need to be positive in order to define a valid probability measure for each subset. Determinants can be viewed as scaling factors of the transformation represented by their matrix. If one of these principal minors is negative, then it means that there exists a vector x for which the transformation $x^\top \mathbf{K} x$ can be negative, which means there exists a negative eigenvalue. Since all principal minors should be positive, it means the kernel has to be positive semi-definite. The requirement that K is positive semi-definite turn out to be sufficient to define a DPP, so when designing a kernel measuring correlation between elements, ensuring this property allows to use the kernel for DPP sampling.

The original definition of DPPs using equation 3.3 is useful for understanding the repulsive property but isn't practical for use, since it only gives the marginal probability of inclusion of any subset A in the sampled set B. In order to work with exact probabilities, that is we want an expression of the probability for each subset A (which can be viewed as the realization of the random variable $\mathbf{A}$), we use a subset of DPPs called L-Ensembles [Borodin and Rains, 2005, Kulesza and Taskar, 2012] (sometimes L-DPP), for which the kernel is called L to differentiate with the original K.

This class of DPPs is still defined through a positive semi-definite kernel L as follows: for any finite subset B of Y,

$$\mathbb{P}[\mathbf{A} = B] \propto \det(L_B), \tag{3.6}$$

where L_B is the matrix defined by L over elements of B. The the normalization constant has a closed form ([Kulesza and Taskar, 2012], Theorem 2.1) given by $\sum_{B \subseteq Y} \det(L_B) = \det(L + I)$, where I is the $N \times N$ identity matrix. We then have an exact representation of $\mathbb{P}[\mathbf{A} = B]$ as:

$$\mathbb{P}[\mathbf{A} = B] = \frac{\det(L_B)}{\det(L + I)}. \tag{3.7}$$

This formulation retains the core repulsive property of DPPs, as we can show by applying this new kernel to the same 2-element set example from 3.5:

$$\mathbb{P}_L[\{i, j\} = B] = \mathbb{P}_L[\{i\}]\mathbb{P}_L[\{j\}] - \Big(\frac{L_{ij}}{\det(L + I)}\Big)^2. \tag{3.8}$$

It can also be shown that the probability $\mathbb{P}[\mathbf{A} \supset B]$ that the random set includes B is exactly $\det(K_B)$, where $K := (L + \mathrm{Id})^{-1}L$.

Building upon this base DPP definition, we now explain how DPPs are integrated in our context. The goal is to produce a maximally diverse set of N future trajectories. To this end, we follow [Yuan and Kitani, 2020a] and define the ground set Y as the finite set of the N predicted trajectories. Intuitively, the overall diversity defined by L over Y reflects into the expected cardinality of the associated DPP. As this expectation reads

$$\mathbb{E}(|\mathbf{A}|) = \mathrm{trace}[\mathrm{Id} - (L_Y + \mathrm{Id})^{-1}], \tag{3.9}$$

see [Kulesza and Taskar, 2012], the expression in the r.h.s. can be used to define the diversity loss for the DSF. This yields:

$$\mathcal{L}_{\mathrm{dpp}}(\hat{\mathbf{S}}_{\mathrm{f}}^{(1:N)}; L) = -\mathrm{trace}[\mathrm{Id} - (L_Y + \mathrm{Id})^{-1}], \tag{3.10}$$

where $Y = \{\hat{\mathbf{S}}_{\mathrm{f}}^{(1)}, \dots, \hat{\mathbf{S}}_{\mathrm{f}}^{(N)}\}$ and L is a kernel to be defined on trajectories. Given two future trajectories $\hat{\mathbf{S}}_{\mathrm{f}}^{(i)}$ and $\hat{\mathbf{S}}_{\mathrm{f}}^{(j)}$ predicted from a same past and present, the trajectory kernel can be simply defined as a spherical Gaussian kernel. This, however, proves insufficient to promote *directional* diversity among the generated trajectories. Hence, we also include in the kernel the angular deviation between the final points of the two trajectories:

$$L(\hat{\mathbf{S}}_{\mathrm{f}}^{(i)}, \hat{\mathbf{S}}_{\mathrm{f}}^{(j)}) = \exp -\alpha(\theta_{ij} + \|\hat{\mathbf{S}}_{\mathrm{f}}^{(i)} - \hat{\mathbf{S}}_{\mathrm{f}}^{(j)}\|_{\mathrm{F}}^2), \tag{3.11}$$

where $\alpha > 0$ is a parameter, $\theta_{ij} \in [0, \pi]$ is the un-oriented angle between segments $(\mathbf{S}_{\mathrm{p}}(T_{\mathrm{p}}), \hat{\mathbf{S}}_{\mathrm{f}}^{(i)}(T_{\mathrm{f}}))$ and $(\mathbf{S}_{\mathrm{p}}(T_{\mathrm{p}}), \hat{\mathbf{S}}_{\mathrm{f}}^{(j)}(T_{\mathrm{f}}))$, and $\|.\|_{\mathrm{F}}$ denotes the Frobenius norm.

3.4.4.2 Quality with a layout loss

The incentive towards diversity proposed in the previous section cannot be left unchecked, lest we generate a set of trajectories that are indeed diverse but not admissible. In order to avoid pathological increase in diversity which would send generated trajectories out of the driving area, we need to ensure their quality, which in our case refers to their admissibility in terms of drivable area. To explicitly control the quality of the forecasted trajectories, we leverage the physical constraints given by the drivable area by introducing a loss term, $\mathcal{L}_{\mathrm{layout}}$, to penalize trajectories predicted out of the drivable area. This binary information is part of the environment bird-eye-view map $\mathbf{M}$ and available in most driving datasets. See figure 3.1 for an example of trajectories drawn on a binary bird-eye-view.

In order to leverage this binary map as a usable diversity loss without 0 gradient almost every-

where, we soften it by applying a Chamfer distance transform on it [Borgefors, 1984] $d_C(A, B) = \sum_{a \in A} \min_{b \in B} D(a, b)$. For each non-zero point a in the set A of non-zero binary map points (i.e. points outside of the drivable area), the distance D to the nearest drivable area (B) point b. For this work we use the standard Euclidean distance as the distance measure.

The soft map $\mathbf{M}_c \in [0, 1]^{H \times W}$, can be seen in figure 3.4 (b).

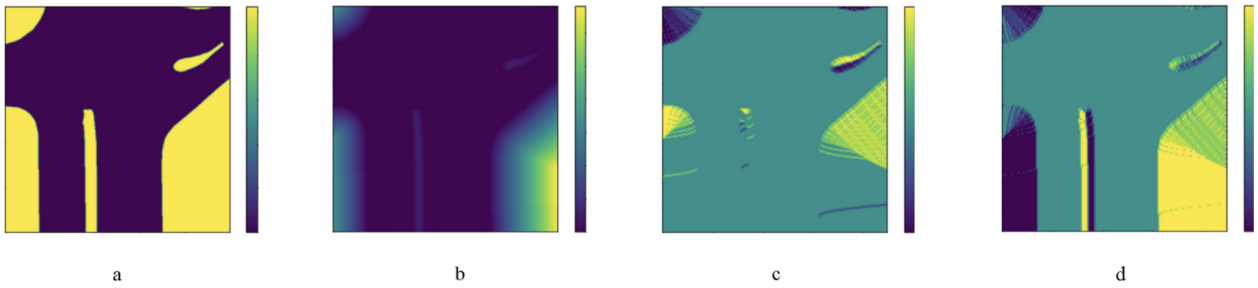


Figure 3.4: **Layout Loss Chamfer map**. From a binary drivable area mask (a), the Chamfer distance map (b) is computed at every point, giving for each point the distance to the closest point in the drivable area. Gradient maps on vertical (c) and horizontal (d) coordinates are then pre-computed for every layout in order to not slow the learning process with redundant calculations.

$\mathbf{M}_c$ allows us to define a differentiable objective with respect to the coordinates of points in a given trajectory. For each point in a generated trajectory, we convert if from output space (relative distance in meters from last past trajectory point) to discrete pixel space and add the Chamfer distance value as a loss, penalizing points that go out of the drivable area. Formally, given $\{\hat{\mathbf{S}}_{\mathrm{f}}^{(i)}\}_{i=1 \cdots N}$ the N future trajectories predicted by our model from an input pair $(\mathbf{S}_{\mathrm{p}}, \mathbf{M})$, our layout loss is defined as:

$$\mathcal{L}_{\text{layout}}(\hat{\mathbf{S}}_{\mathrm{f}}^{(1:N)}; \mathbf{M}_c) = \sum_{n=1}^{N} \sum_{t=1}^{T_{\mathrm{f}}} \mathbf{M}_{\mathrm{c}}(\hat{\mathbf{S}}_{\mathrm{f}}^{(n)}(t)). \tag{3.12}$$

As demonstrated in [Bansal et al., 2018] in the context of imitation learning, providing strong learning signals related to driving rules, such as penalizing off-road driving and collisions, does not work if the predictions are trained to match real-world driving recordings which actually do not include off-road examples. As such, when we generate trajectories that maximize their likelihood under the training data, we cannot make use of such a layout loss because it does not generate enough learning signal. Pairing the inclusion of physical constraints with a diversity-generating mechanism via a gating operation allows us to strike a good balance between diversity and quality, which produces the best results.

3.5 Experiments

Given our proposed architecture and training scheme, we conducted experiments aimed at answering the following questions: (1) How does the addition of a DPP-based training scheme improves the overall diversity? (2) How is the quality of the generated diversity impacted by the layout loss? (3) Is the overall accuracy of the model with respect to the ground truth conserved in experiments on a real-world driving dataset?

3.5.1 Metrics

As discussed in section 2.3, a comprehensive set of metrics is needed to thoroughly evaluate diversity in many regards:

Diversity and admissibility metrics. Measuring diversity can be challenging because diversity isn't a term that is sufficient in itself, which is why several diversity metrics have been proposed to evaluate the diversity. In order to measure diversity with respect to the ground truth trajectory, we use the ratio of average Final Distance Error (FDE) to min FDE, $\text{rF} = \frac{\text{avgFDE}}{\text{mFDE}}$ [Park et al., 2020], which is a measure of the spread of the proposed trajectories relative to the ground truth: a high value indicates a high avgFDE, meaning some predictions are far away from the ground truth, and a small mFDE, i.e, one of these predictions is close to the ground truth.

To measure the spread of the predicted set independently from the ground truth, we use the Average Self Distance (ASD) and Final Self Distance (FSD), introduced in [Yuan and Kitani, 2020a].

In order to add a qualitative assessment of the diversity [Park et al., 2020] proposed the Drivable Area Occupancy (DAO), which measures the diversity in predictions that are in the drivable area, as a way to express whether the proposed set of trajectory adequately covers the available driving area.

It's easy to see that a method providing unbounded diversity regardless of the constraints of the layout would produce high values for the aforementionned metrics. In order to asses whether the predicted set is admissible, we use the Drivable Area Count (DAC) [Chang et al., 2019] defined as $\text{DAC} = \frac{N-m}{N}$, where m is the number of predictions that exit the drivable area (DA).

As discussed in [Park et al., 2020], these metrics can be seen as providing complementary information about diversity: rF, ASD and FSD quantify the diversity in terms of mere spread; DAC only assesses the admissibility of the trajectories in the set; DAO captures a mix of diversity and admissibility, by measuring the spread among admissible trajectories only. DAO and DAC, being directly related to the drivable area, they provide valuable insights into how the proposed trajectories resemble real trajectories. We were not able to reproduce [Park et al., 2020], so we do not report the FSD and ASD metrics as they were not originally evaluated in the paper. This does not impair the

results as DAO and rF provide a good diversity assessment.

Ground-truth metrics. In addition to diversity metrics, we evaluate the accuracy of our method with respect to the dataset's unique ground-truth trajectory, traditionally assessed with an Euclidean distance. Following the existing trajectory forecasting literature [Park et al., 2020, Phan-Minh et al., 2020, Yuan and Kitani, 2020a, Yuan and Kitani, 2020b, Bansal et al., 2018, Rhinehart et al., 2018], we use the minimum Average Distance Error (mADE) and Final Distance Error (mFDE), computing the error on respectively all the points of the trajectory or only the final one.

3.5.2 Experimental setup

Dataset nuScenes [Caesar et al., 2020] is a real-world driving dataset consisting of around 850 driving scenes of 20 seconds each. Splitting the driving scenes in 6 seconds past and 3 second future segments, allowing for fair comparison, we obtain 12852 trajectories, 12256 of which are non-outliers usable ones. These scenarios were recorded in Boston and Singapore, respectively left and right-hand traffic regions, and include more maneuvers and layouts than highway-specific datasets such as [REF HIGHWAY]. Detailed annotations allow for a variety of tasks including trajectory prediction and a "bird's-eye-view" maps containing information such as drivable area and pedestrian crossings is available. However, as a real-world dataset, it offers only a single ground-truth future trajectory for each past trajectory, which makes learning multiple-output prediction difficult.

DIVA setup. Our experiments are conducted with the following parameters: The loss balancing coefficient λ is set to 0.5, the latent dimension d_z to 16 and the past-embedding dimension d_h to 128.

Bird's-eye-view (BEV) maps. The BEV maps represent a 50m by 50m square region in the real world around an agent (25m on each side of the agent, 40m in front of it and 10m behind it). They are encoded as RGB $H \times W$-sized images, where $H = W = 224$. Following prior work, these BEV maps are constructed by superimposing both static scene elements (drivable area, pedestrian crossings, stop signs, lanes) and dynamic ones (other agents), each category being distinctively color-coded. In addition, for each surrounding agent, previous positions are also reported (with time-stamped color coding).

Conditional variational autoencoder (cVAE). The recurrent encoder of the cVAE model is a unidirectional GRU [Cho et al., 2014] with $d_h = 128$ units, so that $\mathbf{h} \in \mathbb{R}^{128}$. The CNN encoder for BEV maps is a ResNet18 [He et al., 2016] with a final layer mapping the output to $d_m = 128$, giving $\mathbf{m} \in \mathbb{R}^{128}$. Finally, the decoder is composed of a unidirectional GRU with $d_h + d_m + d_z = 128 + 128 + 16 = 272$

hidden units, paired with a fully-connected layer to project this result onto the final output trajectory space of $T_{\mathrm{f}} = 6$ future steps.

Diversity sampling function (DSF). The DSF is made of two identical branches comprised of four fully-connected layers of size 512, with batch normalization [Ioffe and Szegedy, 2015] and leaky ReLU activations.

Parameter α. The definition of kernel L in Eq. 4.5 includes a parameter α to improve convergence. As long as this parameter is positive, the kernel remains positive semi-definite and the DPP formulation is valid. Setting this parameter to $\alpha = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\theta_{ij} + \|\hat{\mathbf{S}}_{\mathrm{f}}^{(i)} - \hat{\mathbf{S}}_{\mathrm{f}}^{(j)}\|_{\mathrm{F}}^2)$, to calibrate the kernel around the mean of inner kernel values for a particular set $\mathcal{Y}$, works best and is a method often used in kernel methods [Shawe-Taylor et al., 2004] and particularly in DPPs [Guen and Thome, 2020].

Major model parameters are highlighted in table 3.1. More detail can be found in the accompanied code base [2].

Description	Parameter	Value
Latent dimension	d_z	16
Past embedding dimension	d_h	128
Layout embedding dimension	d_m	128
Kernel scaling factor	α	—
Loss balancing parameter	λ	0.5
Past horizon	T_{p}	12 (6s)
Future horizon	T_{f}	6 (3s)
Number of predicted trajectories	N	12

Table 3.1: **Model parameters**.

3.5.3 Results and discussion

Results. We compare our method to three baselines using a generative backbone: for methods using a cVAE backbone, we compare with [Lee et al., 2017] as a reference for cVAE generative models without any explicit diversification mechanism, and with [Yuan and Kitani, 2020a], a diversity method also using DPPs. Originally tested on a toy dataset for trajectory prediction, we report here the results of [Yuan and Kitani, 2020a] when tested on the real-world dataset nuScenes. We also include a comparison with CAM-NF [Park et al., 2020], a recent method involving a diversification mechanism built upon a Normalizing Flow (NF) attentional backbone with the whole drivable area as

[2] https://github.com/lcalem/DIVA

an equiprobable ground-truth distribution for possible futures. As for other diversity methods, we measure diversity and admissibility for 3s predictions with 6s of past history.

Model	Backbone	mADE ↓	mFDE ↓	DAO ↑	DAC ↑	rF ↑
DESIRE [Lee et al., 2017]	cVAE	1.079	1.844	16.29	0.776	1.717
L2 DPP [Yuan and Kitani, 2020a]*	cVAE	1.148	2.272	13.31	**0.975**	1.891
DIVA	cVAE	**0.942**	**1.449**	**34.99**	0.972	**4.907**
CAM-NF [Park et al., 2020]	NF-A	0.639	1.171	22.62	0.918	2.558

Table 3.2: **Prediction assessment on nuScenes**. Evaluation of quality, diversity and admissibility metrics (computed on $N = 12$ predictions) for 3s forecast by our best model and cVAE-backbone baselines. We also include CAM-NF [Park et al., 2020] for the sake of completeness, even though it has a different backbone, preventing comparisons.
*: Our implementation.

Results in Table 3.2 indicate that our best model, including the DPP loss with a combined angle and Gaussian kernel, has the best performance, improving the diversity both in quantity, as measured with the spread relative to the ground truth (rF), and also in quality. For completeness, we included in our results the mADE and mFDE metrics which measure the precision of the best prediction compared to the ground truth, although the focus of this work is on diversity. These metrics depend mostly on the generative model used during the initial training, which explains the better precision on these metrics of [Park et al., 2020] which has a backbone relying on attention mechanisms and normalizing flows. We use a cVAE backbone and obtain results similar to those of DESIRE and L2 DPP on these metrics, as expected due to the generative backbone being the same. All diversity metrics (DAO, rF and DAC) show a marked increase compared to [Park et al., 2020] despite the simpler backbone, showing the significance of our contribution on diversity.

Ablation study. To analyze the contributions of our architecture and losses, we perform an ablation study, the results of which can be seen in Table 3.3. As a baseline, we start by training a very simple cVAE backbone with the loss given by Equation 4.1, and assess its performance on the predicted trajectories decoded from **h**, **m** and a **z** component sampled from the Gaussian prior.

As expected, the results of the cVAE baseline are relatively mediocre on the quality metrics mADE and mFDE, although consistent with the results of DESIRE, which makes use of a "rank-and-refine" module in addition to the cVAE. Low scores of DAO, rF, ASD and FSD are also expected, since the model fails to diversify the predictions and essentially predicts stacked trajectories that go straight. This outcome also explains well the very high DAC measure, as the prediction almost exits the drivable area. By replacing the Gaussian sampling by a "weak" DSF composed of only one branch ('DSF 1B'), improvements on diversity are seen when training the DSF with the diversity loss ('D') but not

Model	mADE ↓	mFDE ↓	DAO ↑	DAC ↑	rF ↑	ASD ↑	FSD ↑
cVAE	1.374	2.682	10.91	0.975	1.246	0.120	0.165
DSF 1B D	1.152	2.275	13.04	0.975	1.881	0.642	0.872
DSF 1B L	1.383	2.723	5.81	0.977	1.046	0.048	0.058
DSF 2B D	1.018	1.594	35.08	0.917	4.948	2.319	3.033
DSF 2B (D+L)	0.942	1.449	34.99	0.972	4.907	2.142	2.842

Table 3.3: **Impact of each component**. Evaluation of the contribution of each component to the quality and diversity of $N = 12$ predictions over 3s on nuScenes.

when training with the layout loss only ('L'). This is expected as the layout loss does not enforce any diversity constraints. When using our two-branch DSF architecture ('DSF 2B') with an element-wise product to combine $\mathbf{z}_m$ and $\mathbf{z}_p$, significant improvements in diversity occur. If the DSF is trained using the diversity loss only ('D'), diversity scores are at their maximum, at the expense of the DAC metric which shows that some (8.3%) trajectories exit the drivable area as a result of the diversification. Adding the layout loss ('D+L') improves the quality again to levels comparable to the non-diverse baseline cVAE, at the expense of a slight drop in raw spread as indicated by the decrease in ASD and FSD (-3.67% and -5.67% respectively).

Fusion. As discussed subsection 3.4.4, the fusion between layout and diversity encodings $\mathbf{z}_m$ and $\mathbf{z}_p$ is a crucial feature of our model. We compare various fusions in Table 3.4 and highlight that the best results are obtained by the element-wise product, validating the gating hypothesis.

Model	mADE ↓	mFDE ↓	DAO ↑	DAC ↑	rF ↑	ASD ↑	FSD ↑
concat	1.146	2.270	12.293	**0.972**	1.765	0.633	0.858
sum $\bigoplus$	1.007	1.833	28.175	0.935	3.083	1.474	1.881
product $\bigodot$	**0.942**	**1.449**	**34.992**	**0.972**	**4.907**	**2.142**	**2.842**

Table 3.4: **Ablation of the fusion between layout and diversity encodings**. Metrics computed on $N = 12$ predictions over 3s in nuScenes, with three ways to combine $\mathbf{z}_m$ and $\mathbf{z}_p$.

3.5.4 Model analysis

Loss balancing. In figure Figure 3.5, we show the effects of varying the balance between the $\mathcal{L}_{\text{layout}}$ and $\mathcal{L}_{\text{past}}$ loss terms in Equation 3.2. The axes are chosen to be FSD as a measure of diversity quantity (as it measures the spread of the proposed trajectories) and DAC as a measure of diversity quality (as it measures the percentage of proposed trajectories that stay in the drivable area), to best show the tradeoff between quantity of diversity and quality of this diversity when varying λ. At one extreme

$\lambda = 0$, we suppress the DPP loss entirely, resulting in a very low diversity and a very high quality. The other extreme, $\lambda = 1$, zeroes out the layout loss (although we still have both DSF branches in the architecture), yielding, as expected, results similar to the second row of Table 3.3.

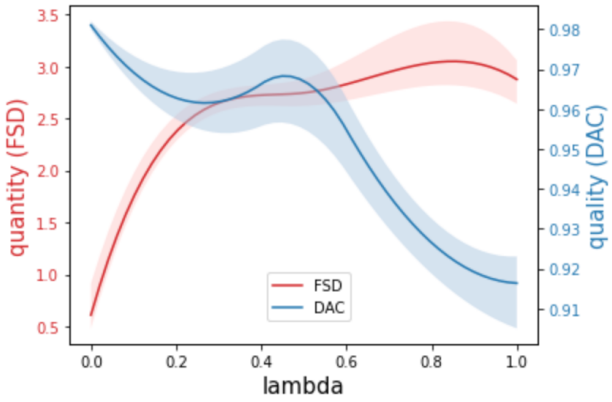


Figure 3.5: **Impact of the balance between quality and quantity losses**. Influence of the weighting parameter in the loss $\mathcal{L}_{\text{dsf}} = \lambda\mathcal{L}_{\text{dpp}} + (1-\lambda)\mathcal{L}_{\text{layout}}$, measured by FSD (red, left axis) and DAC (blue, right axis). For $\lambda = 0$, diversity is suppressed and trajectories stay in the drivable area but have low spread; at $\lambda = 1$, the diversity is maximal at the expense of admissibility.

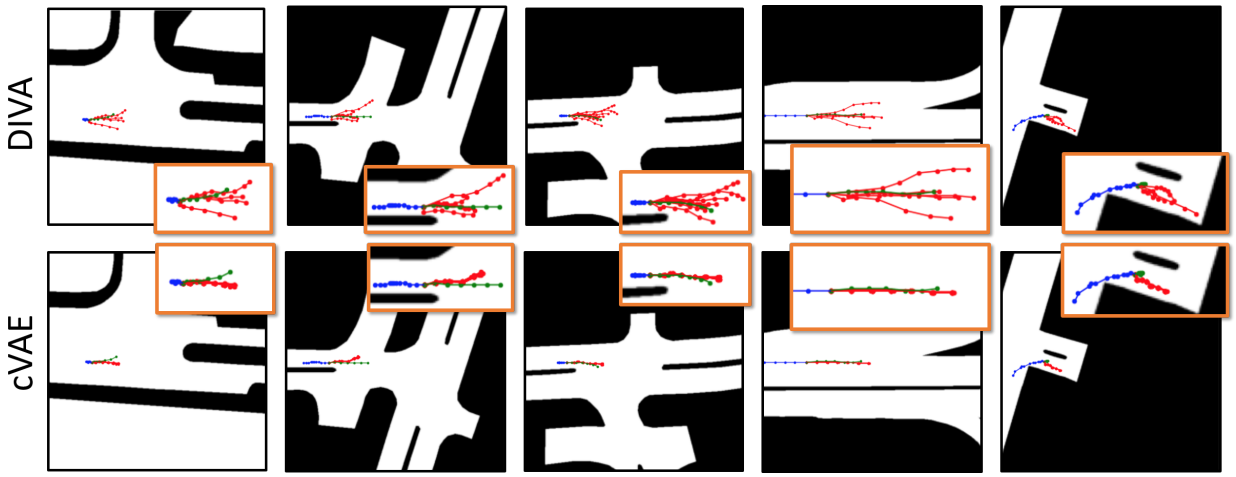


Figure 3.6: **Qualitative results for various scene layouts in nuScenes**. (Top) Results from proposed DIVA model. (Bottom) Results on the same scenes with a simple cVAE, showing a focus on longitudinal diversity (speed) at the great expense of lateral diversity (direction). In each scene: past and future ground-truth trajectories are in blue and green, resp., while predicted future trajectories are in red (best viewed in color).

Qualitative results. As diversity is particularly difficult to assess in numbers only, it is interesting to see how diverse trajectories fare in real-world situations. In figure 3.6, we highlight the diversity

improvements from our model on a variety of scenes, including intersections, straight lines and parking. Recalling that both the cVAE and our method produce $N = 12$ future trajectories, note how the basic cVAE model (lower row), on the same situations, exhibits a mode collapse, sometimes in a direction that isn't the same as the ground truth. The diversity of our model is influenced by both the past trajectory and the layout of the scene, resulting in diverse directions that are plausible in a given situation. Additional qualitative results are also available in Annex A.

3.6 Conclusion

In this chapter, we introduce DIVA, a multi-output forecasting method for predicting diverse yet admissible trajectories. We use a DPP probabilistic model for diversity, and introduce a specific DPP kernel for predicting diverse driving options, leveraging the variety of settings present in the training data. The compatibility of the proposed diverse set with the drivable area is controlled by the inclusion of an admissibility loss independent from the underlying generative model. Quantitative and qualitative experiments on real-world dataset nuScenes confirm the benefit on diversity of the proposed architecture and training scheme, while retaining good accuracy on the sole ground truth trajectory.

Chapter 4

Discovery in the absence of training data

Chapter Abstract

This chapter explores a challenge that goes beyond enhancing diversity in generative models, by leveraging external admissibility constraints in a self-labeling scheme to promote discovery of new minority modes. Traditional generative models often fail to represent the less likely trajectories, leading to a lack of diversity and even less discovery. To address this, we propose a novel approach that goes beyond the constraints of training data to generate trajectories that are admissible but unseen in the training distribution.

Our methodology involves creating a synthetic dataset with multiple future trajectories for each past trajectory in various map layouts, deliberately omitting certain modalities during training. The core objective is to generate trajectories that span the admissible space, including both seen and unseen (in training) trajectories. We introduce a novel training constraint and a self-labeling scheme to facilitate the generation of out-of-distribution trajectories. This approach aims not only to replicate the diversity present in the training data but also to discover new, admissible trajectories.

We present a comprehensive analysis of different training schemes, investigating their learning characteristics to better understand the underlying mechanisms and guide future model designs. Our work raises and addresses critical questions about generating initial diverse proposals, identifying admissible yet out-of-distribution trajectories, incorporating them into the model's learning process, and evaluating the diversity achieved. This research contributes to the broader understanding of generative models, offering insights into how diversity can be more effectively incorporated into these systems.

Contents

4.1 Motivation

The work described in the previous chapter laid out the focus on diversity by designing a method able to generate minority modes represented in the training data, in order to counteract the lack of diversity we can observe in non-diversity focused generative models. In order to achieve diversity, this type of approach have to assume that training data contain in some way the diversity we wish to generate, so that the pre-trained generative model's latent space contains areas (of varying size according to likelihood) for each diverse possibility we wish to generate.

In these diverse trajectory prediction works, likelihood and admissibility are closely linked: the goal is to predict trajectories that are both admissible and more or less likely, in the sense that they are represented in some form in the training dataset. Non-diversity focused methods will predict variations of the more likely trajectory, whereas diversity-focused methods will strive to also generate some trajectories in the less likely regions of the latent space. However, both methods rely on the training dataset as the only source of diversity and assume that the shape of the likelihood distribution is similar to the admissibility distribution. This conclusion leads us to further questioning: for all generative tasks, are admissible predictions the same as likely (as in represented in the dataset) predictions? For trajectory prediction, the answer seems to be clearly no: if we have a crossroads with three possible roads, and our dataset contains only trajectories going forward, likely trajectories will all go forward, leaving left and right trajectories with likelihood 0. However, these alternative trajectories are clearly admissible, despite being unlikely in the sense of the training dataset. The

discrepancies that exist between the admissible distribution, the likely distribution and the generated distribution are highlighted in figure 4.1.

This line of questioning gave rise to a new and more challenging question that we started tackling in this work: how to generate a diverse set of elements, including elements that are admissible but unlikely under the training distribution of a generative model? In other words, how do we generate trajectories that are never seen during training?

It is unlikely that we will be able to generate data that are admissible to humans without any kind of learning signal, as a model eventually needs to have some information. However, most models across many tasks mostly rely on leveraging the information contained in the training dataset, simply because that's the only available data. In conjunction with previously developed diversity methods that capture the diversity exhibited in the training data, our approach also tries to leverage information not contained in the training dataset, but in order to expand the generated distribution, not constrain it.

As driving datasets contain only one future trajectory for each past trajectory, we first created a synthetic dataset representing several future trajectories for each past trajectory, in different map layouts. This first step is necessary to have full control on the training and evaluation, to assess whether the task is at all possible. During training, one modality is completely omitted, like all trajectories going left. The goal is to be able to generate trajectories that cover the admissible space (figure 4.1 bottom), while being different from the trajectories seen during training. Of course, we also want to generate those in the set of predicted trajectories, in addition to the discovered, completely new trajectories.

Generating admissible data far from the training data distribution isn't a standard task, and we propose a method to validate its feasibility. In our method, the first challenge is to be able to generate out-of-distribution trajectories, to start exploring out of the training distribution. To this end, the first training constraint we impose is to train the whole model (encoder, latent code generator and decoder) in an end-to-end fashion. Contrary to DIVA where the latent code generator (3.2) can be trained on top of a pre-existing trained encoder-decoder backbone, discovery necessarily needs the decoder to be trained at the same time, because it needs to learn how to decode latent codes that will represent new trajectories. If a diversity component is added on top of a frozen pre-trained decoder, it is not possible to generate samples that go out of the support distribution of the decoder part of the VAE model, as we investigate in this chapter. In order to conduct this investigation and understand better the constraints of our diversity model that prevent discovery, we also build a toy model suitable for this exploration and support the findings. Second, in order to control the discovery and not constrain the decoder too much with the past trajectory (which holds a large amount of predictive information), we cut the direct link between the encoder's outputs (embeddings) and the decoder's inputs.

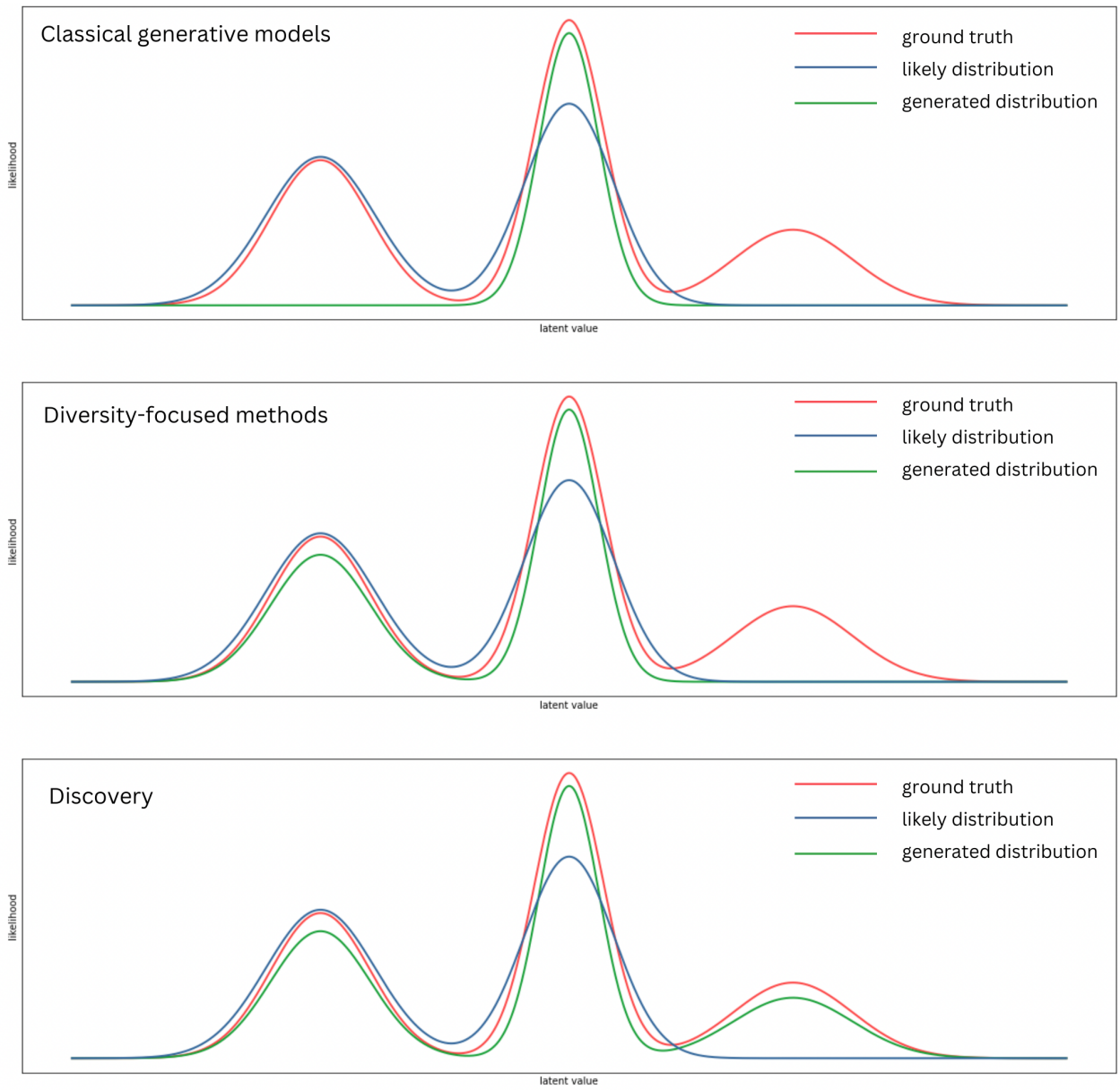


Figure 4.1: **Illustration of the discrepancies between admissible, likely and generated distributions**. For a one-dimensional idealized example with multi-modal ground truth, we illustrate the discrepancy that exists between the admissible (ground truth, in red) distribution that ultimately represents the truth, the likely distribution (in blue) that represents the training data, and the different generated distributions (in green) under different model approaches. (top) for traditional generative models, all samplings of a latent code z are variations around the majority mode, adding some noise. (middle) in diversity-focused methods, the goal is to find the areas of the latent space that is formed by the training data that correspond to different modalities. (bottom) in the discovery process, we aim at expanding the latent space itself to modalities never seen in the dataset.

Equipped with these two constraints and a self-labeling scheme, we can train a model on the aforementioned synthetic trajectory dataset, in order to generate new (discovered) trajectories. In addition to the two modes present in the data (forward and right trajectories), we also want to generate admissible left trajectories that is absent from the training dataset, where all trajectories going left have been removed.

In summary, this line of work gives rise to several challenging questions:

- How do we generate initial diverse proposals?
- How do we identify these admissible but out-of-distribution trajectories?
- How do we include them in a self-labeling fashion and encourage further discovery of admissible trajectories?
- How do we evaluate such a diversity?

We provide elements of answer to these questions in the present chapter, along with an analysis of the learning characteristics of different training schemes to better understand the mechanisms at play and inform new design ideas.

4.2 Designing an experimental setup

The discovery problem laid out thus far is an interesting but challenging one, out of the boundaries of classical benchmarks and tasks in computer vision and trajectory forecasting that have a well defined practical target, such as segmentation and ground truth trajectory matching.

In order to step out of the hand-wavy explanation of the task at hand, the first challenge we have to tackle is very practical:

> How do we design experiments to test for successful discovery in the context of trajectory forecasting?

We step into this problem without knowing if a solution exists, so the first step is to design a proof-of-concept experiment that will allow us to assess whether the discovery process can happen in any situation. In order to achieve that, we need a fully controlled test environment, so we start by designing a suitable toy dataset for this experiment.

4.2.1 Problem formulation

In the applicative context of this thesis, namely trajectory forecasting, we have the same elements to work with than the previous diversity task. Figure 4.2 gives a visual reminder of these elements. As inputs, we have the ground truth binary bird-eye view (BEV) map $\mathbf{M}$, representing the layout of the scene, in addition to the past trajectory $\mathbf{S}_{\text{p}}$. The layout is scaled to a standard 224 x 224 pixels size and the past trajectory covers the past horizon T_{p}.

The goal, as in the diversity task, is still to produce N future trajectories $\mathbf{S}_{\text{f}}$, representing the future possible trajectories. The difference between the diversity and discovery task lies in the training dataset. For the diversity task, all modalities (*e.g.*left, straight and right trajectories) are present in the training dataset, but in imbalanced quantities creating a majority mode. The task is then to find a method to retrieve these minority modes at prediction time, constrained by the layout of the scene.

Figure 4.2: **Task input and outputs**. The binary drivable area is represented as the black-and-white bird eye view, the past trajectory in blue and the N future trajectories in red.

For the discovery task, the training dataset has to be different: if we integrally suppress one modality across all layouts so that the model never learns it from the training examples, we are left with a discovery task that has to leverage other means to produce the desired modality. As a side note, not all modalities can be equally removed in order to qualify for an extrapolation task. As highlighted in figure 2.4, removing the middle modality while retaining both left and right trajectories, even if it seems to be removing the same "amount" of data, devolves the task in a simpler interpolation (Recombination-to-range in Montero et al.'s terms) task.

Having made this distinction clear, we can define the discovery task for trajectory forecasting as having a modality removed from the training set, but not a modality that can be interpolated from other training modalities. For our setting, it means removing all trajectories going either left or right, but not a training dataset that removes all straight trajectories but keeps both left and right trajectories, as it would be interpolation and not extrapolation.

4.2.2 Synthetic dataset

Following [Yuan and Kitani, 2020a] experiments showing diversity on a synthetic trajectory dataset involving one cross-shaped layout, we start by building a similar-looking layout filling a 224x224 pixel image, to conform with the model capacity.

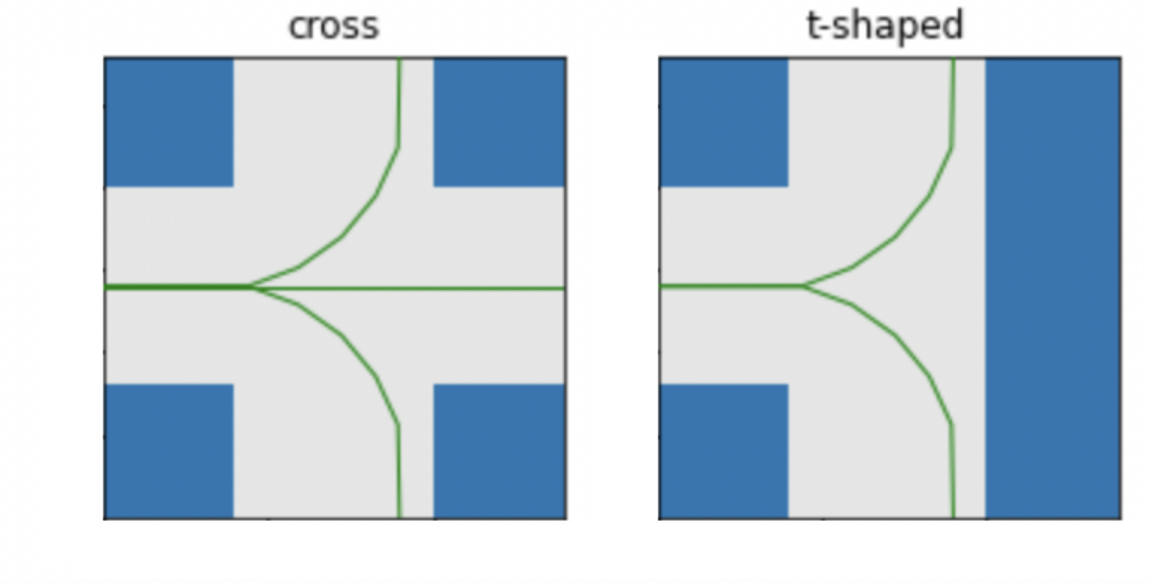
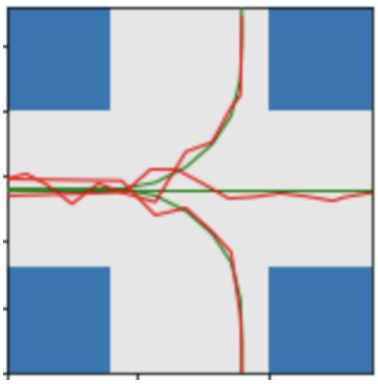


Figure 4.3: **Toy dataset layouts and ground truths**. In these two different layouts of the toy dataset, the canonical ground truth trajectories share a common past trajectory at the leftmost part of the trajectory and then cover the possible future modes of the layout.

After the layout, we create the canonical ground truth trajectories (in green on Figure 4.3) by hand. These trajectories are composed of 8 to 10 segments, spanning the overall ground truth directions.

From these canonical trajectories, generating ground truth trajectories that are variations of these and can be used as a training set isn't as straightforward as adding Gaussian noise to the ground truth points. As can be seen in figure 4.4, sampling each point in the trajectory from independently and identically distributed Gaussians does not result in realistic trajectories.

Figure 4.4: **Gaussian noise trajectory creation**. Creating a real-looking trajectory from canonical ground truth points isn't as straightforward as adding Gaussian noise to the points.



To overcome this limitation, we create a way to pick n waypoints for a trajectory, then fill the inside points with either a linear oversampling or a a curved oversampling, depending on the trajectory specifications. That way, we can sample $n = 3$ waypoints from a canonical trajectory: one at the beginning, one at the turning point and the endpoint, then interpolate smoothly between these selected points in order to create more realistic-looking trajectories. The results of this interpolation created trajectories can be seen in Figure 4.5.

During experiments, we willfully omit all trajectories going in the left direction, while retaining

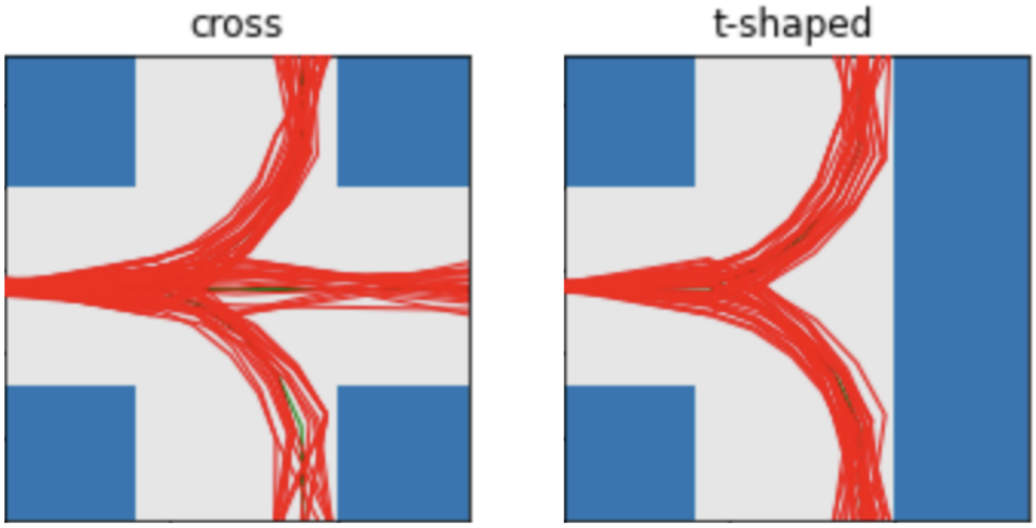


Figure 4.5: **Generated smooth test trajectories**. In these two different layouts of the toy dataset, the ground truth trajectories share a common past trajectory at the leftmost part of the trajectory and then cover the possible future modes of the layout.

the left-going trajectories for the test set, upon which will the recall be evaluated. This omission is key to the discovery problem. Putting even one left-going trajectory in the training dataset, or trying a simpler setup of having left trajectories in one layout and right trajectories on another layout, would fundamentally alter the very nature of the task, turning the problem into a combinatorial generalization one.

4.2.3 Metrics

In order to equip our model with the capability to discover unseen trajectory modalities, we need a test. By deliberately omitting left trajectories during training and then challenging the model to generate them, we aim to evaluate its true generative capacity. To quantify the model's success for discovery, we have identified the following metrics:

Recall Recall is an essential metric for understanding how many of the generated trajectories align well with the actual, albeit unseen, left trajectories. Specifically, we look at trajectories with an Average Displacement Error (ADE) below 2 meters with the ground truth trajectory as successful discoveries. Higher recall indicates the model's aptitude in uncovering missing trajectory modalities

Closest distance While recall provides a broader view, it relies on a "hit-or-miss" metric, and we also want a precise measure of the model's accuracy. Hence, for each generated trajectory, we compute the distance (ADE) to the closest ground truth left trajectory. A lower "best distance" is indicative of a more accurate prediction.

Top 10 best distances To further understand the model's consistency in generating accurate trajectories, we look at the top 10 best distances for each generated trajectory to the ground truth left path. This metric provides insights into the spread and consistency of the generated paths relative to the true paths.

The combination of these metrics provides a sufficient evaluation of our model's capacity for discovery. While recall tells us how often our model gets it right within an acceptable margin, the best distances give us exactly how good is the accuracy of these predictions. The best distance tells us if whether the model is able to discover at all the missing modality, while the best 10 metric tells us whether this discovery was a fluke or not. With these metrics in hand, we can ensure the model is not just mimicking seen data but is truly capable of creative generation, a task that models usually don't handle [Montero et al., 2022].

4.3 Proposed approach

The challenge we are tackling here lies in finding an approach to effectively generate modalities that are absent from the training data but still crucial for comprehensive model performance. Addressing this, we propose a novel solution to effectively generate and harness this missing modality. The first challenge is the generation of diversity, ensuring a broad spectrum of potential modalities. The subsequent step incorporates a selection function, picking the most pertinent elements to reintroduce them through a self-supervised learning framework. Central to our methodology is an encoder-decoder architecture. Rather than a sequential training approach like DIVA, our strategy necessitates simultaneous training of both the decoder and the diversity-promoting mechanism. This simultaneous approach counters the potential bias that might arise if the decoder were to be trained in isolation, leading to a more robust and efficient solution.

In order to discover new trajectory directions, the proposed approach focuses on two critical steps: first, the model needs to be able to generate samples that go slightly out of the training distribution, and second we need to identify these interesting samples to use them in a self-supervised fashion. In the following sections we detail these two components in greater detail.

4.3.1 One-step model

The first step to discovery is the generation of suitable trajectories. As a necessary but not sufficient condition, finding the right architecture for the problem at hand is the first task to tackle:

Which model architecture is suitable for the generation of out-of-distribution yet acceptable trajectories?

The relatively simple dimensionality of the trajectory generation limitation of the generative discovery problem allows to work in an encoder-decoder framework with latent space that we can visualize and manipulate quite easily.

cVAE As a model generating vehicle trajectories, we use a simple generative model composed of an encoder-decoder architecture in the form of a conditional Variational Auto Encoder (cVAE), in order to offer a latent space sampling option in which to guide and constrain generation. As a sanity check, we first train a basic cVAE on the synthetic dataset described above to test for discovery capabilities. If we train a basic model with the vanilla cVAE objective for trajectory prediction:

$$L_{\text{cvae}}(\phi, \theta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{S}_\text{p},\mathbf{M})}[\log p_\theta(\hat{\mathbf{S}}_\text{f}|\mathbf{z}, \mathbf{S}_\text{p}, \mathbf{M})] - \beta \text{KL}(q_\phi(\mathbf{z}|\mathbf{S}_\text{p}, \mathbf{M})\|p(\mathbf{z})), \tag{4.1}$$

as expected, no diversity is exhibited, and even less so for trajectories that could be used for discovery (see figure 4.6 for qualitative results). In the cVAE framework, q_ϕ is the encoder, who produces latent codes $\mathbf{z}$ from past trajectory embeddings $\mathbf{S}_\text{p}$ and layout embeddings $\mathbf{S}_\text{p}$. p_θ is the decoder, that takes the latent code $\mathbf{z}$ along with the past trajectory embedding $\mathbf{S}_\text{p}$ and the layout embedding $\mathbf{M}$, to produce a set of future trajectories $\hat{\mathbf{S}}_\text{f}$. β is a term controlling the strength of the regularization, absent in the original VAE framework [Kingma and Welling, 2014] but added soon after in an effort to better control [Higgins et al., 2017] the regularization term. Some applications can benefit from a better regularized smooth latent space, but in our case, adding even large values of β didn't result in any significant improvement on diversity (see Appendix B.1 for results).

cVAE - no KL In this setting, the KL divergence term is a regularization term which pushes the latent space to be smooth and continuous [Higgins et al., 2017] (see Figure 4.7). A first approach can be to remove this term to try to create areas in the latent space that are not successfully reconstructed by the decoder, in order to be able to pick latent codes that aren't fully mapped by the decoder to a known trajectory.

Removing this term, for our problem, did not result in a particularly non-smooth latent space

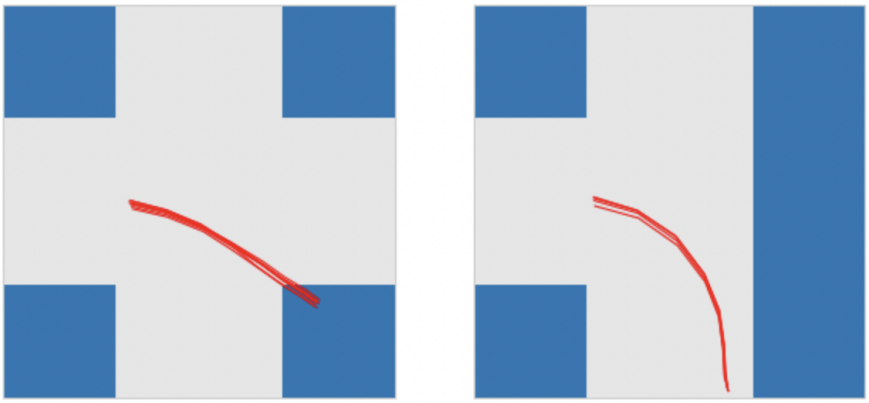


Figure 4.6: **Generation outputs on the synthetic dataset for a vanilla cVAE model.**. As the model is trained on the synthetic dataset training set, containing only straight and right-going trajectories, the cVAE model generated distribution exhibits no diversity at all. In the case of the t-shaped layout (right), only one trajectory is seen, which is reflected in the predictions that closely match the training trajectories. In the cross layout (left), two modes are seen during training and the generation is the average trajectories of these two modalities. This discrepancy highlights that the cVAE rightfully takes into account the layout in its prediction, but nothing pushes for diversity.

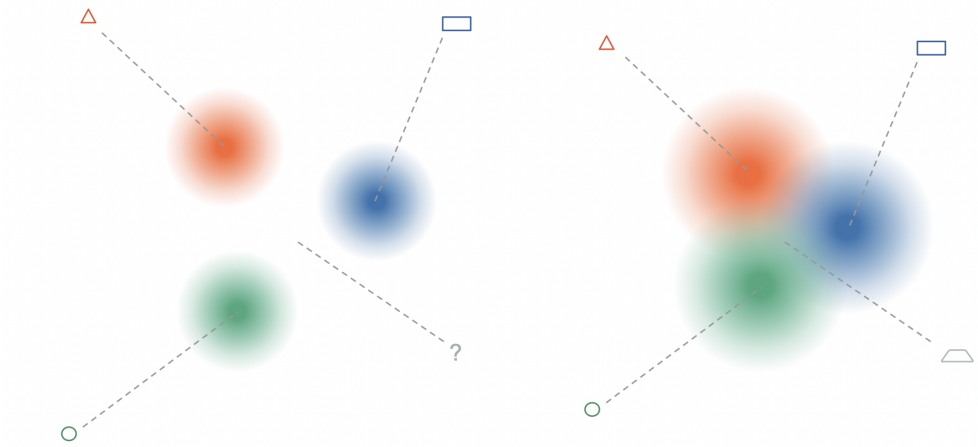


Figure 4.7: **Theoretical effect of the KL-divergence term on latent space**. In a regularized space (right), one can expect a smooth interpolation between modes. By removing the regularization, we hope the latent space can contain regions where the decoder is unable to generate a trajectory matching a known mode (left).

from where we could sample out of distribution trajectories to be used for pushing the boundaries of the generator. In this setting, the decoder is simply too powerful and adapts to even a non well-behaved manifold by always generating the majority mode (see figure 4.8). This behavior, which is the opposite of what we want for diversity and discovery, can be explained by looking at the loss

function (see eq. 4.1) without the KL component, which turns it into a generic auto-encoder loss function, focused solely on minimizing the reconstruction error. Since the right-going trajectories are overall the dominant trajectories, the default behavior is to always predict a right-going trajectory, as it is the most straightforward way to minimize the reconstruction error.

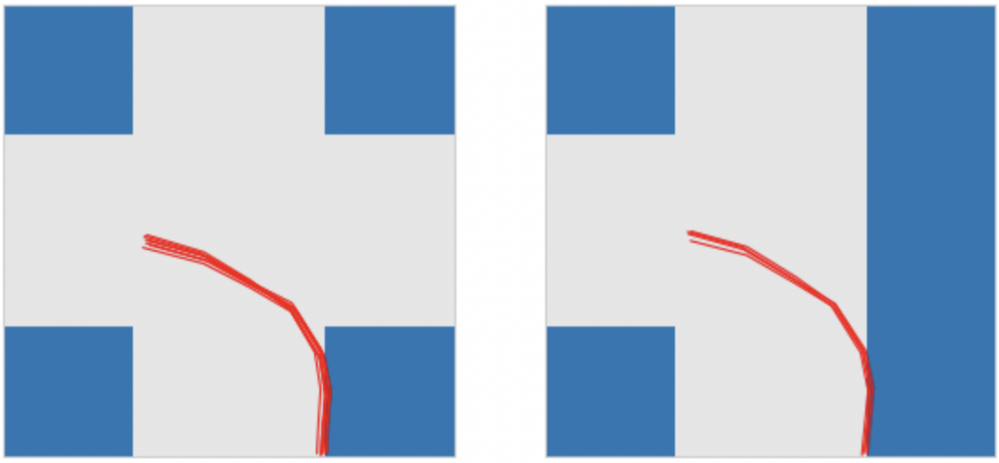


Figure 4.8: **Qualitative results for a no-KL cVAE**. Removing the KL-divergence regularization term does not result in out-of-distribution trajectories but prompts the decoder into generating the same output for every latent variable.

DIVA Adding an explicit output space diversity loss, such as the one presented in the previous chapter, DIVA, seems like a straightforward way to ensure diversity, as we have seen. However, we need to find whether it is sufficient in itself to ensure samples diverse enough to go from diversity to discovery. As shown in figure 4.9, adding the diversity component produces a more spread-out multimodal distribution, but the results are still skewed towards the cVAE's decoder right-going bias.

End-to-end model As we have seen, any two-step approach where the cVAE is trained before the diversity module, has a major drawback. Pre-training the cVAE backbone model on the missing-modality dataset, locks the decoder in a state where it is too biased to produce samples that deviate from the bimodal training distribution.

In order to relieve the constraints of the decoder and allow for the decoding of trajectories that differ from the training distribution, we try an end-to-end training scheme, whereby we train all components at the same time: encoder, decoder, and diversity sampling model. As shown in figure 4.10, we now fully retrieve the bimodal training distribution.

Since this model has been trained end-to-end without a cVAE with the KL component, we want to see if the latent space contains regions that can be decoded in trajectories that go outside the bimodal

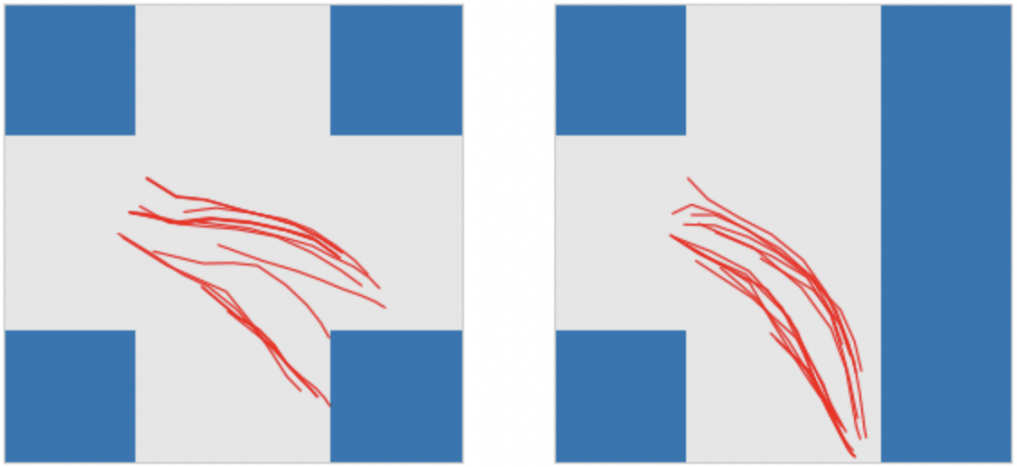


Figure 4.9: **Qualitative results for DIVA on synthetic data with missing modality**. The addition of DIVA's specific diversity component upon the pre-trained cVAE base indeed improves the diversity of the generated distribution but doesn't prevent the heavy bias towards the majority mode (in this case right-going) trajectory.

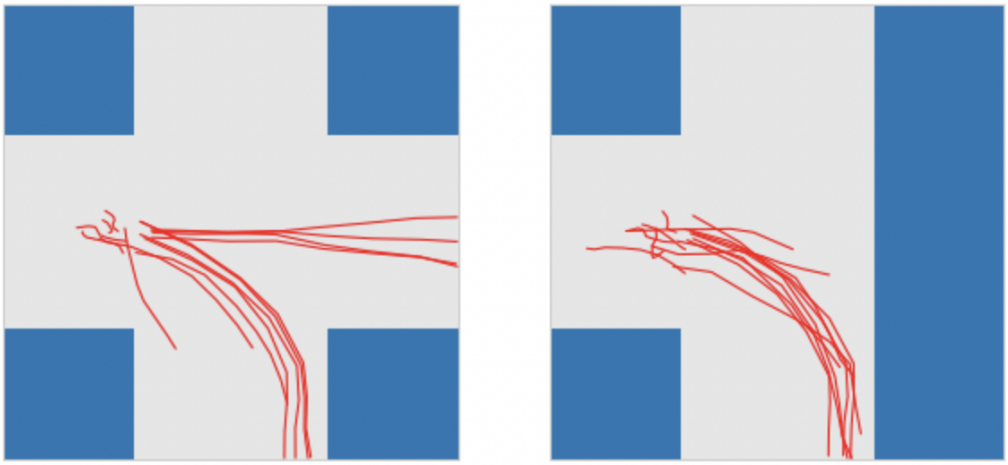


Figure 4.10: **Qualitative results for End-to-end DIVA on synthetic data with missing modality**. Training the decoder at the same time as the diversity component allows for a more balanced result by removing the frozen decoder's bias of DIVA.

distribution.

In order to check for this behavior, we first gather the distribution of latent codes during training and extract the two principal components and their boundary values, $z_1 \in [z_1^{min}, z_1^{max}]$ and $z_2 \in [z_2^{min}, z_2^{max}]$. We artificially construct an array of latent codes $z = [z_1, z_2]$, interpolated between the minimum and maximum value of each component. We decode each z using the decoder part of the model to create the trajectories shown in figures 4.11 and 4.12.

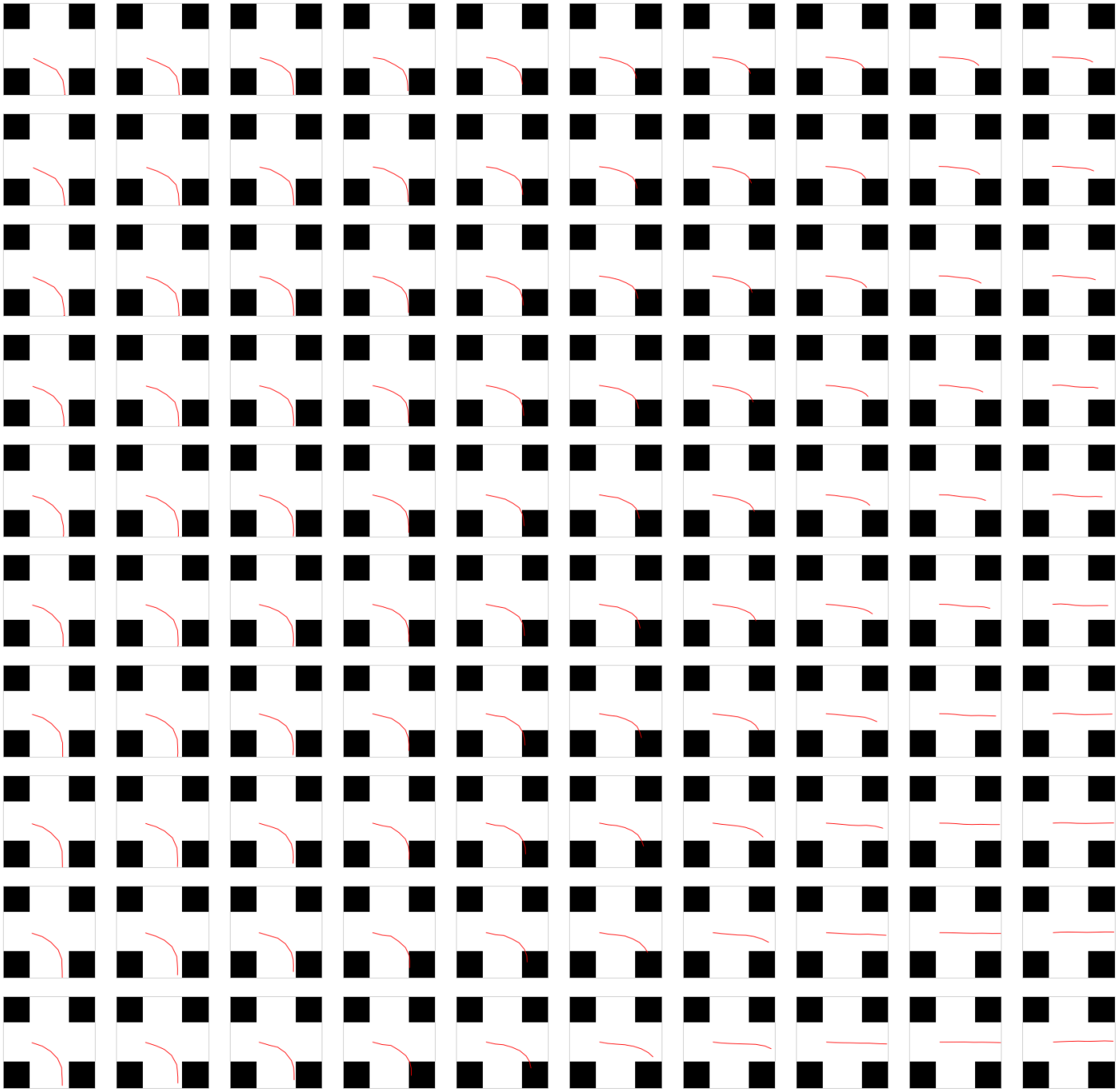


Figure 4.11: **Smooth interpolation between modalities for cross-shaped intersection**. Even in the absence of KL-divergence term in the cVAE loss, a pretrained decoder manages to interpolate smoothly between the two modalities seen in the training dataset. The figure has been created in 2D using a representation of latent codes in 2-dimensions (vertical and horizontal axes representing the two z components)

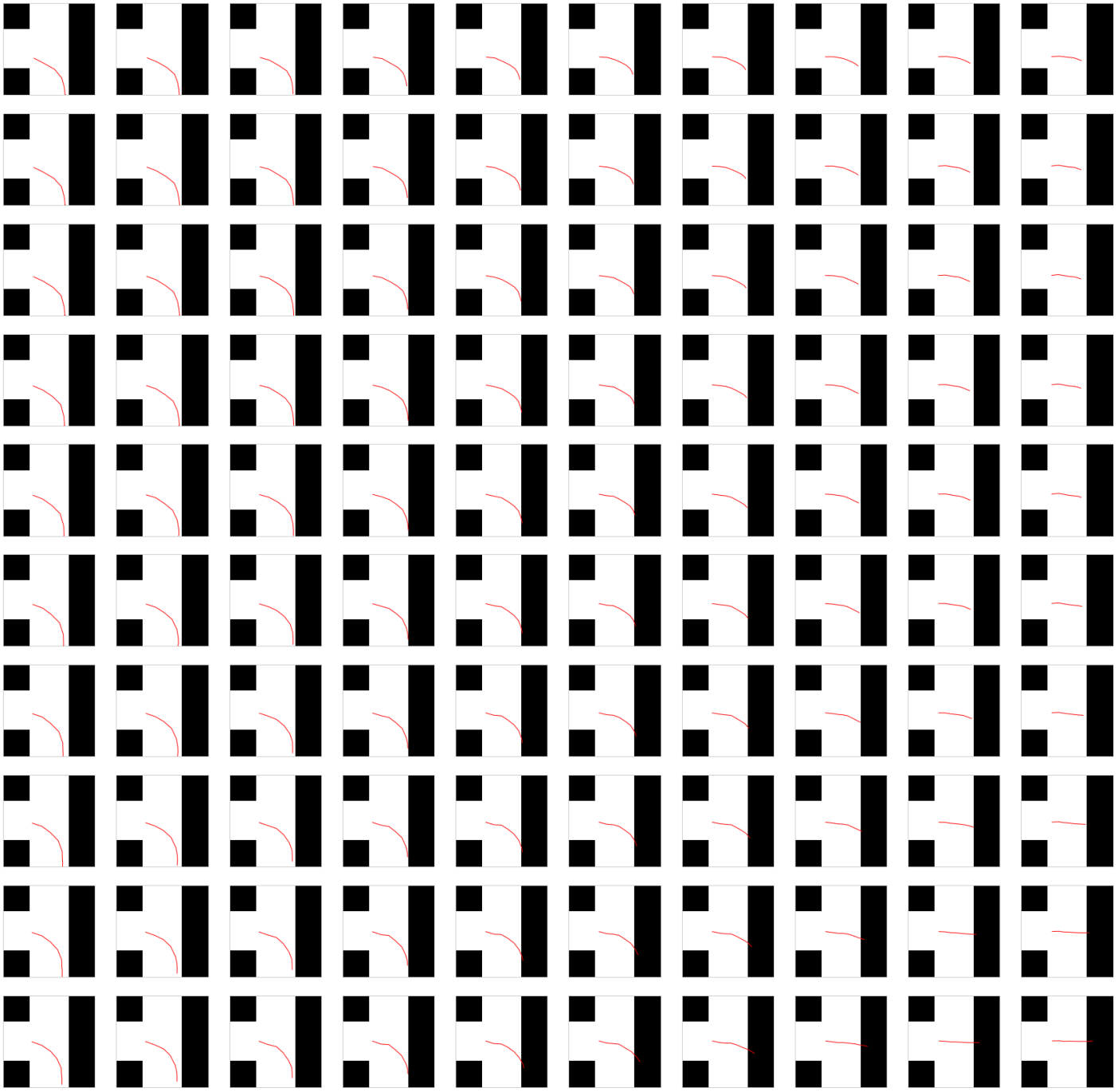


Figure 4.12: **Smooth interpolation between modalities for T-shaped intersection**. Even in the absence of KL-divergence term in the cVAE loss, a pretrained decoder manages to interpolate smoothly between the two modalities seen in the training dataset. The figure has been created in 2D using a representation of latent codes in 2-dimensions (vertical and horizontal axes representing the two z components)

Even though the latent space is less constrained, it does not naturally provide the expected exploration results. If the decoder is powerful enough to match reconstructions for all latent codes in the boundaries of the latent distribution seen in training, maybe we can extrapolate with latent values out of the training distribution. Removing the hyperbolic tangent non linearity at the end of the process sampling z, so the value isn't bounded, we tried to sample latent codes outside of their initial distribution and decode them to see if the decoder would fail to reconstruct a meaningful trajectory. Aggregated results for the cross layout are shown in Figure 4.13. Even for latent codes outside of the training range, the decoder plateaus and maps the unseen latent code to the closest training code. For straight trajectories, we can see a slight left movement but overall the trajectory remains straight and doesn't change endpoint. Additional visualisations for the extrapolation generation are available in Annex B (figures B.2 to B.9).

Allowing for the decoder to be trained during the diversity promoting process allows for a greater malleability of the decoder, which is a necessary but not sufficient condition for generating out of distribution suitable trajectories. In order to do that, we need to add diversity promoting mechanisms, which we will cover in the next section.

4.3.2 Diversity-promoting mechanisms

Architectural bottleneck In order to understand the out-of-distribution generative capabilities we are uncovering, here is a quick reminder of how a regular cVAE model is trained for the trajectory prediction problem (for more detailed explanations see section 2.2.3):

$$L_{\text{cvae}}(\phi, \theta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{S}_\text{p},\mathbf{M})}[\log p_\theta(\hat{\mathbf{S}}_\text{f}|\mathbf{z}, \mathbf{S}_\text{p}, \mathbf{M})] - \text{KL}(q_\phi(\mathbf{z}|\mathbf{S}_\text{p}, \mathbf{M})\|p(\mathbf{z})). \tag{4.2}$$

In the canonical cVAE-based model, we want to find the true conditional distribution $p(\hat{\mathbf{S}}_\text{f}|\mathbf{S}_\text{p}, \mathbf{M})$ of the future trajectories $\hat{\mathbf{S}}_\text{f}$ given the conditioning past trajectory $\mathbf{S}_\text{p}$ and layout embedding $\mathbf{M}$). The cVAE framework models this conditional distribution through a generative process involving a latent space from which latent codes $\mathbf{z}$ can be sampled, which captures the underlying stochasticity of future trajectories. As depicted in the diagram in figure 4.14 (a), the model consists of an encoder $q_\theta(\mathbf{z}|\mathbf{S}_\text{p}, \mathbf{M})$ which infers a distribution over the latent variables from the conditioning inputs $\mathbf{S}_\text{p}$ and $\mathbf{M}$, and a decoder $p_\phi(\hat{\mathbf{S}}_\text{f}|\mathbf{z}, \mathbf{S}_\text{p}, \mathbf{M})$, which generates the predicted trajectories from the sampled latent variables along the conditioning inputs.

It has been shown in experiments from the previous section that the decoder is the main blocker for generating trajectories that are still trajectories but out-of-distribution enough to push the boundaries of the generative distribution. It has also been shown, in [Ben-Younes et al., 2022] for a Trajectron++ [Salzmann et al., 2020] model and in [Xu et al., 2023] for ViP3D [Gu et al., 2023] and

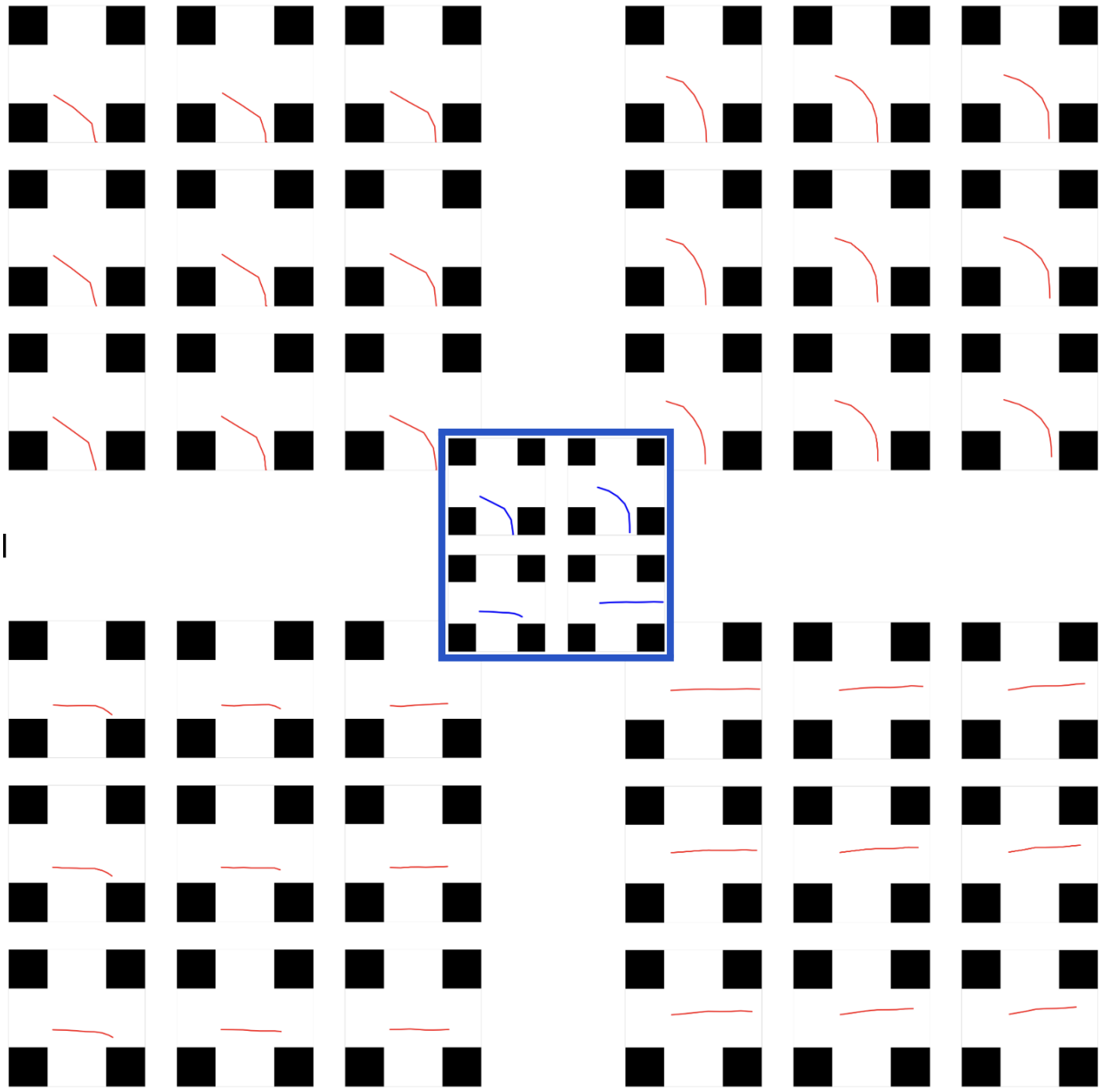


Figure 4.13: **Extrapolation outside the learned range of learned latent codes**. Using a pretrained cVAE baseline, decoding elements outside the pretrained latent code range results in smooth generation of known trajectories, consolidating the hypothesis of a decoder too powerful to generate out-of-distribution trajectories. The middle trajectories (in blue) are the generated trajectories for the corresponding known boundaries of each latent code dimension, for reference.

UniAD [Hu et al., 2023] models, that the past trajectory conditioning information is often informative enough to predict the future trajectory without even considering the layout information.

For these reasons, we propose to remove the connection between conditioning information $\mathbf{S}_\text{p}$, $\mathbf{M}$ and the decoder p_ϕ (see diagram in figure 4.14 (b)). This creates a bottleneck at the latent space level that we hope to leverage for our out-of-distribution generation problem.

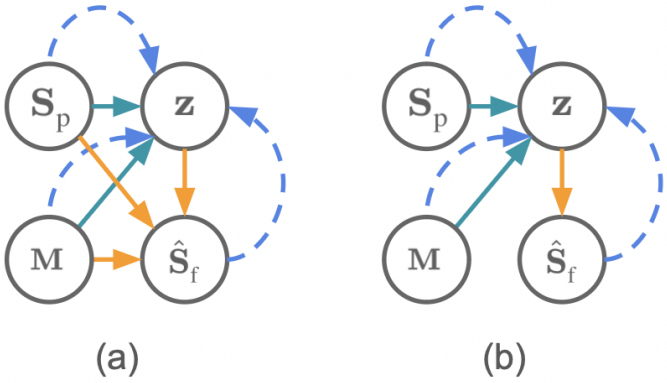


Figure 4.14: **Architectural bottleneck**. By removing the link between the conditioning information and the generated distribution, we increase the generative power of the learned latent distribution z

The decoder model, formerly trained with the following loss without the KL term:

$$\mathcal{L}_\text{rec}(\phi, \theta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{S}_\text{p},\mathbf{M})}[\log p_\theta(\hat{\mathbf{S}}_\text{f}|\mathbf{z}, \mathbf{S}_\text{p}, \mathbf{M})], \tag{4.3}$$

now becomes

$$\mathcal{L}_\text{rec}(\phi, \theta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{S}_\text{p},\mathbf{M})}[\log p_\theta(\hat{\mathbf{S}}_\text{f}|\mathbf{z})]. \tag{4.4}$$

This gives the latent code $\mathbf{z}$, as the sole input for decoder $\log p_\theta$, a much more decisive role in the generation. Instead of being merely a Gaussian noise addition to the majority mode prompted by the past conditioning, $\mathbf{z}$ becomes the primary driver of reconstruction, giving any diversity mechanism working in the latent space much more latitude for promoting diversity.

Diversity Loss function The previous chapter's work encouraged diversity via a DPP-inspired loss $\mathcal{L}_\text{dpp}(\hat{\mathbf{S}}_\text{f}^{(1:N)}; L) = -\text{trace}[\text{Id} - (L_Y + \text{Id})^{-1}]$ which kernel $L(\cdot, \cdot)$ was computed using the inter-trajectory distance in the output space, along with their final point angle θ and a scaling factor α:

$$L(\hat{\mathbf{S}}_{\mathrm{f}}^{(i)}, \hat{\mathbf{S}}_{\mathrm{f}}^{(j)}) = \exp -\alpha(\theta_{ij} + \|\hat{\mathbf{S}}_{\mathrm{f}}^{(i)} - \hat{\mathbf{S}}_{\mathrm{f}}^{(j)}\|_{\mathrm{F}}^2). \tag{4.5}$$

Our goal here is different. While we do still want to promote diversity in the output space, we also want to generate trajectories that are out of the regular distribution. Doing so in the output space with the same loss as DIVA manages to promote the diversity within the constraints of the training distribution, but not outside of it, as shown in the example of figures 4.9 and 4.10.

Instead of promoting the diversity in the output space, working the diversity from within the latent space can offer more control and generality. Since the training of both the diversity promoting component and the decoder is simultaneous, we can leverage the initial weakness of the decoder to create out-of-distribution elements while it still can.

Staying within the DPP framework, proven effective to model the negative correlations, we propose to use a kernel pushing all the latent codes $\mathbf{z}$ as far as possible, while still being constrained by the reconstruction loss (see section 4.3.5 for the final reconstruction loss used in the model). The diversity promoting loss thus becomes:

$$\mathcal{L}_{\mathrm{dpp}}(\mathbf{z}^{(1:N)}; L) = -\mathrm{trace}[\mathrm{Id} - (L_Y + \mathrm{Id})^{-1}], \tag{4.6}$$

with the following kernel:

$$L(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}) = \exp -\alpha(\|\mathbf{z}^{(i)} - \mathbf{z}^{(j)}\|_{\mathrm{F}}^2). \tag{4.7}$$

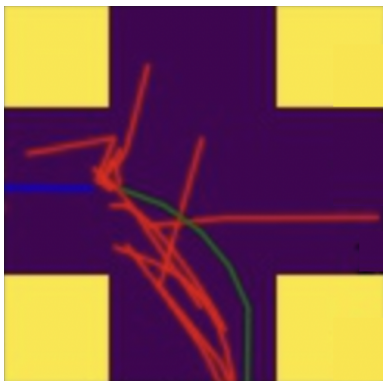
4.3.3 Prediction selection pseudo labeling

Having a diversity-generating model that is able to generate ever-so-slightly out of distribution trajectories that are admissible is a necessary but not sufficient block. Indeed, once these are generated, we need a method to automatically select and leverage these predicted trajectories and identify the most promising ones in order to guide the model further and build a latent space matching the admissible distribution in addition to the training distribution.

In order to expand the capacity of the model beyond the training distribution, we can carefully select the trajectories predicted by our model that deviate from the training distribution just enough to be interesting. We need to identify trajectories that match two characteristics. First, the trajectory needs to exhibit trajectory-like behaviour. An admissible polyline that doesn't look at all like a vehicle trajectory isn't useful (see figure 4.15). Second, the trajectory needs to deviate from the training distribution towards unseen yet admissible modalities.

The balance between these two desirable characteristics is delicate. However, if we manage to generate and identify these, we can reintegrate them in the training dataset as pseudo-labeled trajectories in order to expand the generative distribution.

Figure 4.15: **Examples of admissible yet undesirable trajectories**.



Throughout our experiments, we have found that gradually pushing the generative distribution towards unseen left trajectories is possible, by reinjecting the trajectories closest to the left-going trajectories during learning, in a pseudo-labeling fashion.

Given the fragile nature of the balance between generating trajectories that are visually admissible and the need to generate trajectories that go outside the training distribution, we have found that a cautious pseudo labeling strategy is warranted for discovery. In order to avoid a catastrophic inclusion of undesired trajectories in the training dataset, that makes the training diverge and not achieve any meaningful result even on known modalities, we have found that two major training components are warranted:

First, the pseudo-labeling is turned on only after the first epoch, to allow a warm-up period for the decoder to learn to reconstruct trajectory-like elements. While most trainings are able to converge to discovery without this warm-up, we found that a small portion of the training runs failed without the warm-up.

Second, the optimal number of new examples to add for each batch has been found to be 1 new pseudo-labeled example per 32-element batch. Again, most trainings would converge if 2 or more are added, but including only 1 sample is more reliable.

Using this pseudo-labeling training scheme, we are able to expand the generative distribution. Figure 4.18 shows the evolution of generated trajectories that are closest to the left ground truth over time.

4.3.4 Cross-Attention weighting of latent codes

To enhance the decoder's understanding of potential trajectory directions, we aim to leverage spatial information to modulate the latent codes. This is achieved through executing a cross-attention mechanism between the latent codes Z and the spatial information l_{spatial}, as depicted in Figure 4.16.

We start with the original latent code matrix $Z \in \mathbb{R}^{(BS,N,d_z)}$, with BS denoting the batch size, N the number of predicted future trajectories (12 in most of our experiments), and d_z the dimension of the latent code, also 12 in most of our experiments.

Omitting the batch size dimension for clarity, $Q \in \mathbb{R}^{N,d_q}$ is obtained from $Z \in \mathbb{R}^{(N,d_z)}$ by

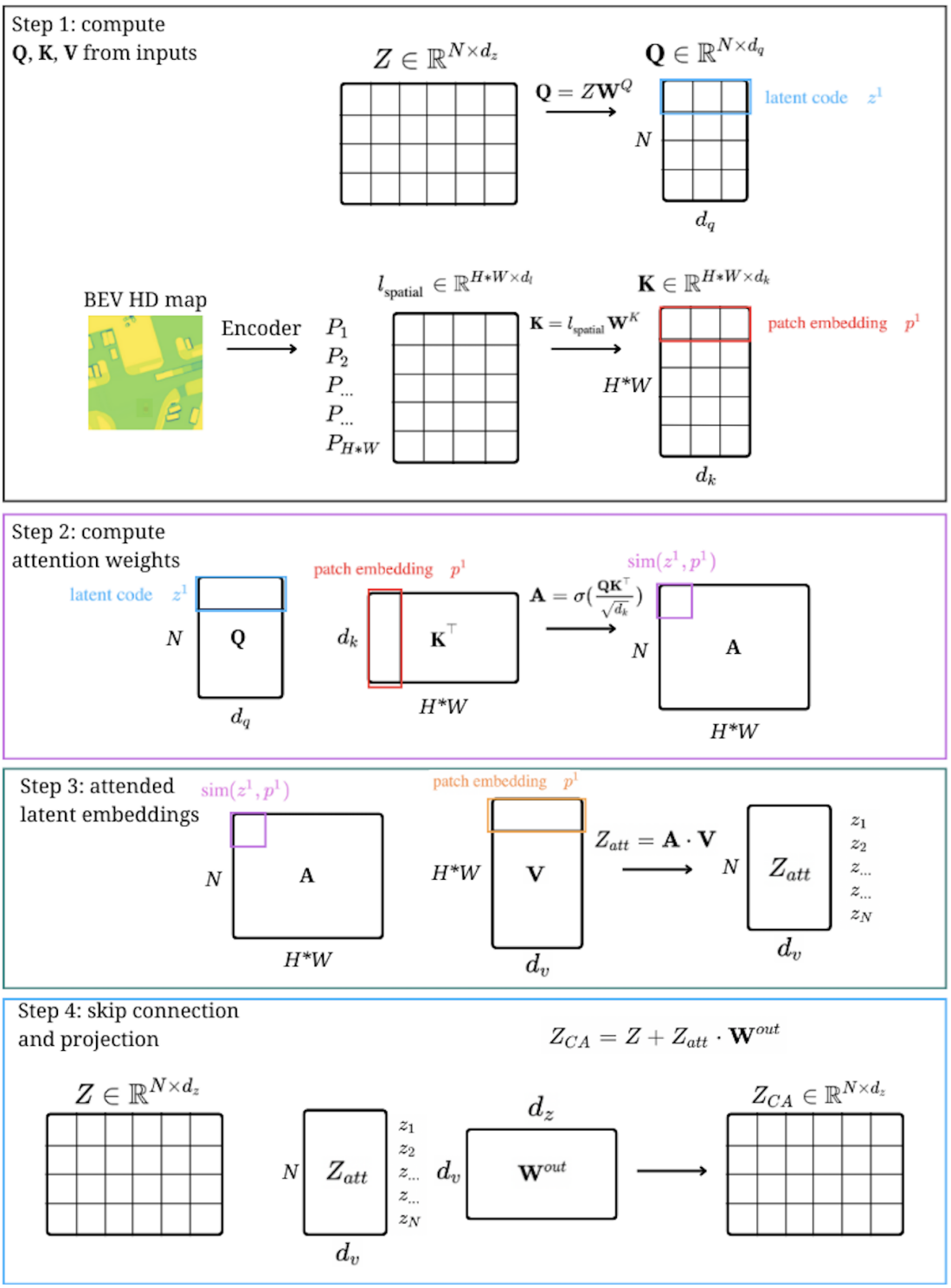


Figure 4.16: **Weighting of latent codes by spatial information via cross-attention**. Detailed cross-attention steps between original latent codes Z and spatial information l_{spatial}.

projecting with a weight matrix $W^Q \in \mathbb{R}^{d_z,d_q}$. Similarly, $K \in \mathbb{R}^{HW,d_k}$ and $V \in \mathbb{R}^{HW,d_v}$ are obtained from $l_{spatial}\mathbb{R}^{HW,d_l}$ with weight matrices $W^K \in \mathbb{R}^{d_z,d_k}$ and $W^V \in \mathbb{R}^{d_z,d_v}$, respectively. A summary of these shapes (including batch size) is provided in table 4.1 for ease of use.

Quantity	Dimensionality	Example	Comment
l_{flat}	(BS, ?)	(32, 128)	Layout embedding
l_{spatial}	(BS, ?, H, W)	(32, 256, 14, 14)	Spatial BEV embedding
Z	(BS, N, d_z)	(32, 12, 12)	Latent code matrix
Q	(BS, N, d_q)	(32, 12, 64)	Query from Z
K	(BS, HW, d_k)	(32, 196, 64)	Key from l_{spatial}
V	(BS, HW, d_v)	(32, 196, 64)	Value from l_{spatial}
A	(BS, N, HW)	(32, 12, 196)	Attention weights
Z_{CA}	(BS, N, d_z)	(32, 12, 12)	Attended latent code matrix

Table 4.1: **Shapes involved in the computation of Z_{CA}.** Summary of the shapes of the different matrices used in the computation of the weighted latent codes matrix Z_{CA}.

From Q, K and V, we compute attention weights $A \in \mathbb{R}^{(BS,N,HW)}$ using the standard cross attention formula [Vaswani et al., 2017] $A = \text{softmax}(\frac{QK^\top}{\sqrt{d_k}})$. The attended latent embeddings Z_{att} are obtained by multiplying the previously computed attention weights A by the V matrix (step 3 in figure 4.16) and the final attended latent code matrix $Z_{CA} \in \mathbb{R}^{(BS,N,d_z)}$ is then obtain by projecting back Z_{att} into the original dimensions of $Z \in \mathbb{R}^{(N,d_z)}$ through an appropriately-sized weight matrix $\mathbf{W}^{out}$ then using a skip connection with the original Z in order to avoid drifting too much from the original latent codes. The resulting latent codes are then used in the discovery pipeline as previously.

4.3.5 Final model

In order to summarize the architectural explorations leading to a working model for the discovery task of interest, we present in this short section the selected architectural choices and losses that we made to advance the results and understanding of the discovery task. The architecture of our discovery model is summarized in figure 4.17:

The model is encoding the input BEV layout $\mathbf{M}$ in an embedding $\mathbf{m}$ and the past trajectory $\mathbf{S}_{\text{f}}$ to embedding $\mathbf{h}$ through the encoder q_ϕ. Both embeddings $\mathbf{m}$ and $\mathbf{h}$ go in their respective Diversity Sampling Function (DSF) branch to produce N semi latent codes $\mathbf{z}_m$ and $\mathbf{z}_h$, which are combined through an element-wise product to produce N final codes $\mathbf{z}$. The latent codes are the only inputs to the decoder p_θ, generating N future trajectories $\hat{\mathbf{S}}_{\text{f}}$.

Now how do we train that model? The objective for discovery are three-fold: first, we need to generate trajectories that actually look like trajectories. This is achieved through a reconstruction loss. Second, we need an incentive for diversity, with a diversity loss. And of course third, we need

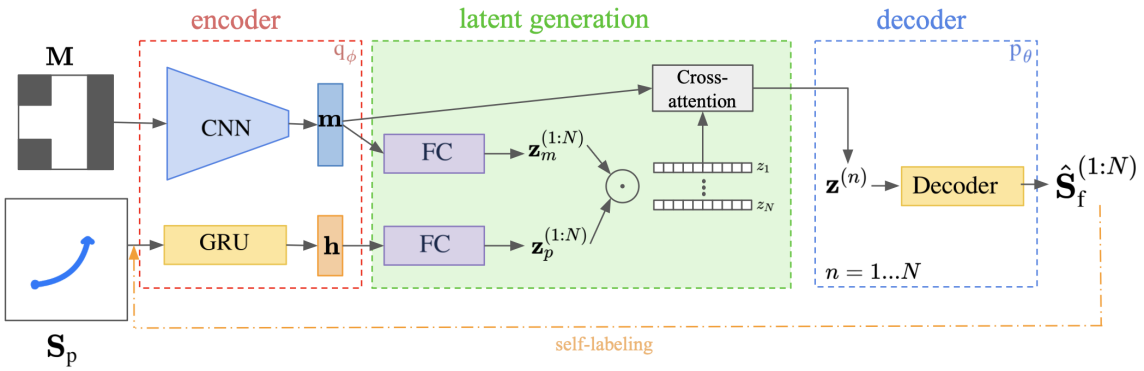


Figure 4.17: **Discovery model for trajectory generation**. General architecture for the discovery model, divided into three logical parts: on the left (red) the encoding of both spatial and past trajectory inputs, in the middle (green) the generation of N latent codes with cross-attention with the spatial embedding, and the rightmost part (blue) representing the generation of future trajectories from latent codes.

those diverse and novel trajectories to be admissible, hence the need of an admissibility loss. The overarching loss equation is thus, unsurprisingly, the following:

$$\mathcal{L} = \mathcal{L}_{rec} + \mathcal{L}_{div} + \mathcal{L}_{adm}. \tag{4.8}$$

Reconstruction loss The reconstruction loss goal is to provide the decoder with enough learning signal to train it to be able to reconstruct trajectory-like elements, while not constraining it too much to allow diversity and discovery to emerge. For the reconstruction, likeness to a trajectory is the main objective. However, finding a way to describe the intrinsic qualities of a trajectories is a pointless task: if we find a function to describe the characteristics of a trajectory, we could use it to generate trajectories in the first place. We then use the traditional log-likelihood associated with the predicted trajectory $\hat{\mathbf{S}}_\text{f}$ as the reconstruction loss:

$$\mathcal{L}_\text{rec}(\phi, \theta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{S}_\text{p},\mathbf{M})}[\log p_\theta(\hat{\mathbf{S}}_\text{f}|\mathbf{z})]. \tag{4.9}$$

However, the reconstruction objective is fundamentally at odds with the diversity objective, since we use the ground truth trajectories to ultimately derive the error between the predicted and ground truth trajectory. In order to avoid over-constraining the decoder with a reconstruction loss forcing all trajectories to resemble ground truth trajectories, we use a softer version of the signal. Instead of backpropagating the reconstruction signal through all N branches generating the N trajectories, we backpropagate only in the branch that produced the trajectory closest to the ground truth. That way, some reconstruction signal is sent in the decoder, but it is faint enough to avoid undermining the

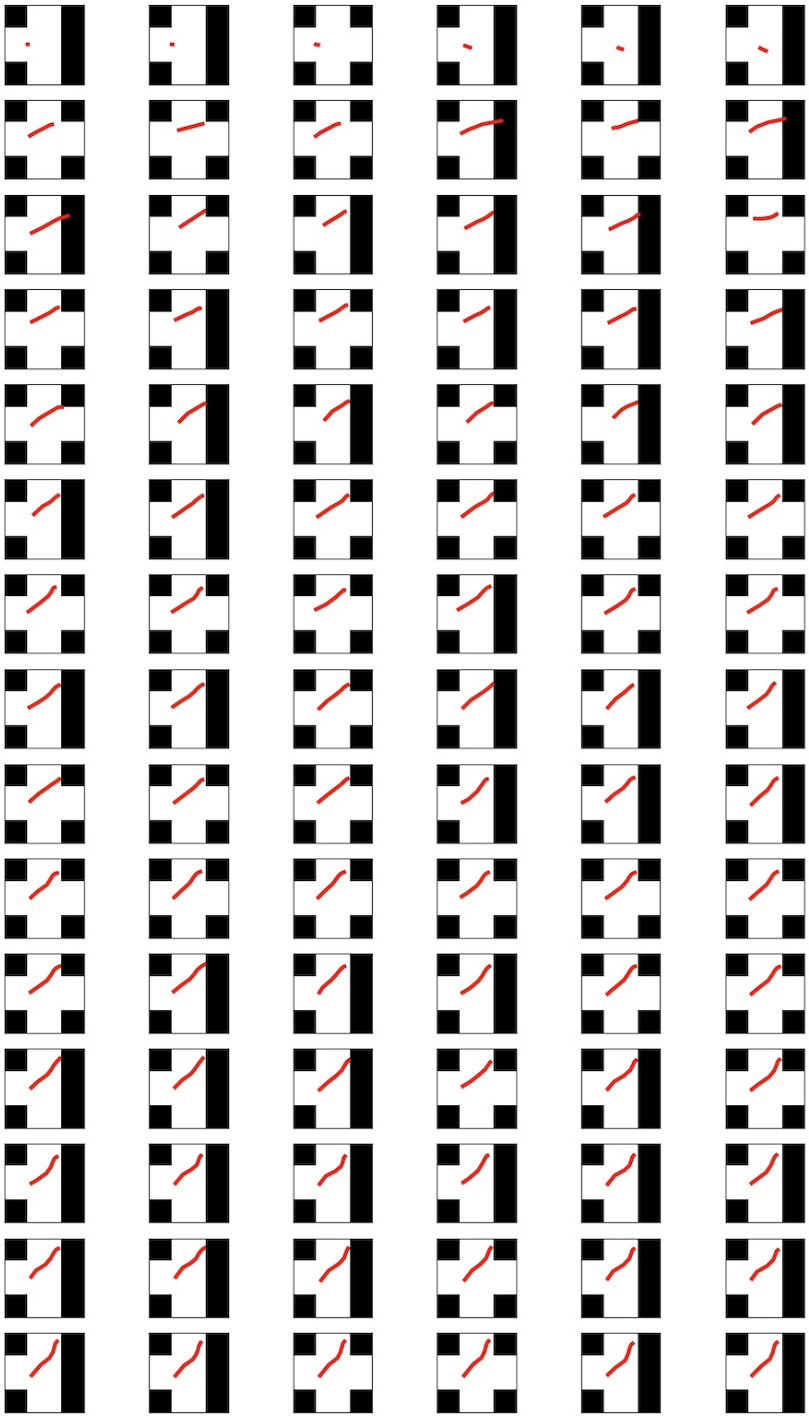


Figure 4.18: **Evolution of generated trajectories closest to left ground truth over training time**. Evolution using a pseudo-labeling of size 1 per 32-image batch. Each row denote an epoch in ascending order, and for each row a number of examples is shown.

diversity generation.

Admissibility loss The admissibility constraint, as mentioned before, incentivizes generated trajectories to stay within the bounds of the drivable area, via a differentiable mechanism explained in section 3.4.4.2. The associated loss,

$$\mathcal{L}_{\text{layout}}(\hat{\mathbf{S}}_{\text{f}}^{(1:N)}; \mathbf{M}_c) = \sum_{n=1}^{N} \sum_{t=1}^{T_{\text{f}}} \mathbf{M}_{\text{c}}(\hat{\mathbf{S}}_{\text{f}}^{(n)}(t)), \tag{4.10}$$

corresponds to the summation over all $T_{\text{f}} \times N$ predicted trajectory points of the value of the differential Chamfer map $\mathbf{M}_c$. This map has a value of 0 for any point inside the drivable area, and increasing positive values for points the further they are from the drivable area.

Diversity loss The interesting part of the total loss is the way discovery is created. Having the latent codes $\mathbf{z}^{(1:N)}$ be the only input to the decoder gives a single point where to optimize for diversity and discovery. It is done by maximizing the distance between all N latent codes within the same set of N to ensure maximal diversity for the generated trajectories. The loss, based on the DPP kernel L described in eq. 4.7, is as follows:

$$\mathcal{L}_{\text{dpp}}(\mathbf{z}^{(1:N)}; L) = -\text{trace}[\text{Id} - (L_Y + \text{Id})^{-1}]. \tag{4.11}$$

4.4 Results

4.4.1 Reconstruction loss

As stated in the previous sections, our model leverages several losses in order to carefully balance between generating trajectories that do look like trajectories and generating diverse enough trajectories that can be used to expand the generated distribution.

Even though the reconstruction loss is fairly straightforward, it is still at odds with the diversity objective. One challenging part in the construction of the discovery model was finding the right balance between the reconstruction loss providing the signal for the decoder to learn how to generate trajectory-like elements, and the diversity loss pushing the decoder to generate trajectories from little known parts of the latent space.

To achieve a balance between constraint and freedom for the decoder, we have explored several strategies to synergistically include both the reconstruction and diversity losses. As we found out, adding them both for every head, even with different weights, didn't produce interesting results

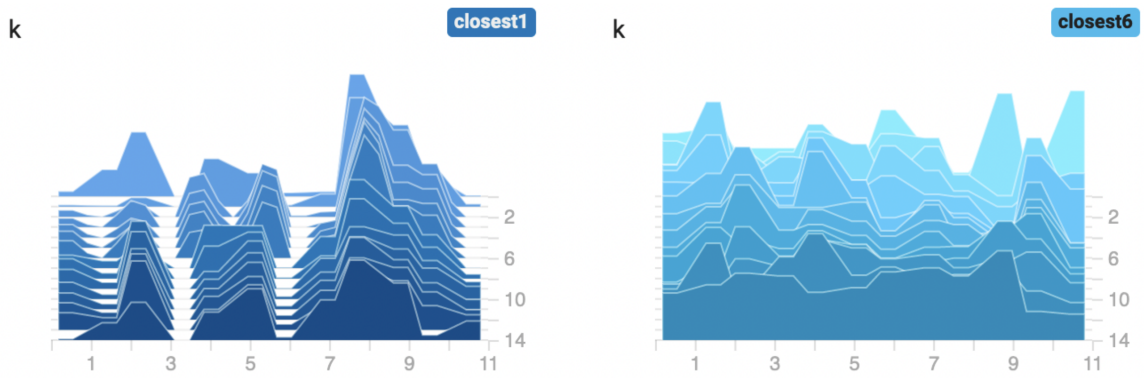


Figure 4.19: **Selected heads for reconstruction loss branch selection after training**. Horizontal axis represents the of the branch selected out of the $N = 12$ total branches for each generated set of trajectories, vertical axis represents the training epoch: furthest and lighter-colored distributions correspond to early epochs and the latest is shown at the foreground of the figure. We show that by selecting only one head (left), some heads become specialized in the reconstruction of trajectories closer to the training distribution.

because those objectives are fundamentally optimizing for opposed desirable aspects of the generated trajectory set. In the final model, we chose to backpropagate the reconstruction loss only in the closest trajectory, in order to let the other heads less constrained and be able to generate more diverse trajectories; figure 4.19 highlights the head selection quantities between two strategies: propagating the loss in only one head corresponding to the generated trajectory closest to the ground truth (left), or in the 6 closest trajectories closest to the ground truth (right). As $N = 12$ in our experiments, it is equivalent to giving the reconstruction signal half of the time.

These results show that naturally some prediction heads will receive more reconstruction than the others, indicating that some heads could specialize in generating trajectories that are closer to the training distribution, while some others could be generating trajectories away from it. Figure 4.20 presents an aggregated reconstruction across multiple batches of trajectories generated solely by heads that did not receive any direct signal from the reconstruction loss in the 1-closest scenario (see Figure 4.19 left). This indicates that the decoder was capable of generalizing the trajectory-like nature of the generated elements, owing to its inductive bias and the influence of partial reconstruction loss.

4.4.2 Discovery and diversity quantitative results

As outlined in section 4.2.1, the presented problem formulation allows for a proof-of-concept framework to answer the question: **is it possible to generate admissible trajectories without any examples in the training data?**

Given the architecture presented in the previous section, we now present the quantitative results

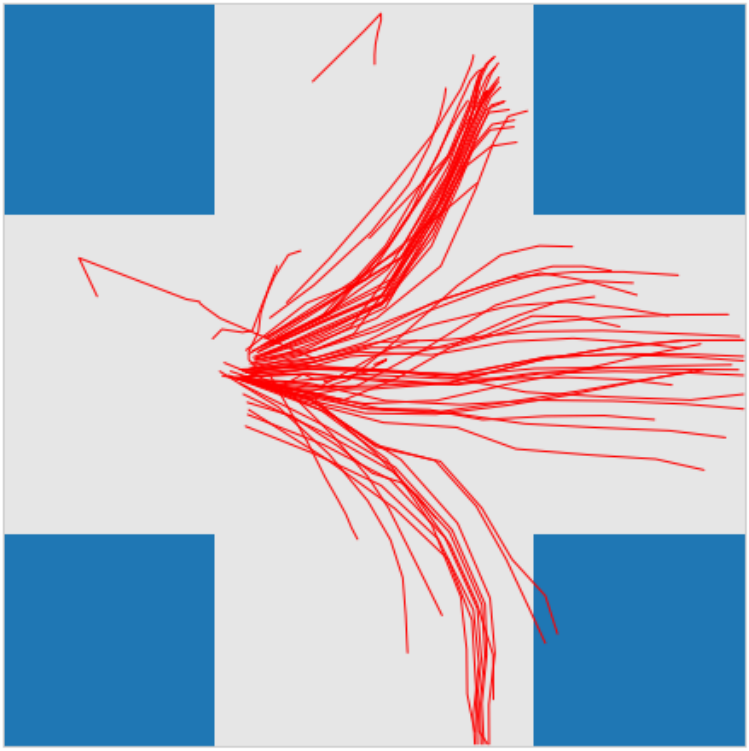


Figure 4.20: **Generated trajectories for non-reconstruction heads**. Example trajectories generated during training from heads that received no reconstruction loss: apart from rare artifacts, there is enough inductive bias in the decoder to generate smooth-looking trajectories for each direction, even for heads that never receive a direct reconstruction loss.

we have under the proposed evaluation framework. To the aforementioned question, we are happy to report that "yes" is an acceptable answer. We detail the quantitative results for this exploration work under the metrics presented in section 4.2.3 in table 4.2.

Selection	best_1	best_{10}	recall
Baseline	3.96	4.64	0.5
Oracle SL	1.05	1.07	0.97
CA Oracle	0.70	0.72	0.97

Table 4.2: **Oracle selection discovery results**.

Selection	mADE	mFDE	rF	DAC	ASD	FSD
Baseline	1.559	3.060	2.399	0.993	3.287	4.705
Oracle SL	0.674	1.247	5.866	0.955	3.819	6.139
CA Oracle	0.697	1.324	5.214	0.938	3.301	5.689

Table 4.3: **Oracle selection accuracy and diversity results**.

Among methods that generate diversity, it is insightful to check whether the diversity of the proposed discovery methods is still better. Intuitively, discovery should encapsulate diversity, and at the very least discovery should not impair the diversity of the generated distribution. Table 4.3 shows the assessment on the diversity metrics used for the diversity method presented in section 3.5.1.

While not the primary focus of the task, the accuracy of the closest trajectory to the ground truth (measured by mADE and mFDE) is better in our discovery methods, owing perhaps to the greater number of parameters of the model. Although the quality of the diversity is slightly degraded by the DAC measure, the diversity metrics, rF, ASD and FSD are greatly improved, more significantly so in FSD as the final point comparison better reveals the broader final point distribution allowed by successful discovery of new modalities.

For these results, the baseline model refers to a model that is successfully able to capture the bimodal nature of the training data, as presented in 4.3.1, such that it already exhibits a diversity representative of the multimodal nature of the training distribution.

4.4.3 Qualitative results

On the first two canonical layouts (cross and t-shaped) figure 4.21 shows that overall the generation of left-going trajectories is working, with or without cross-attention.

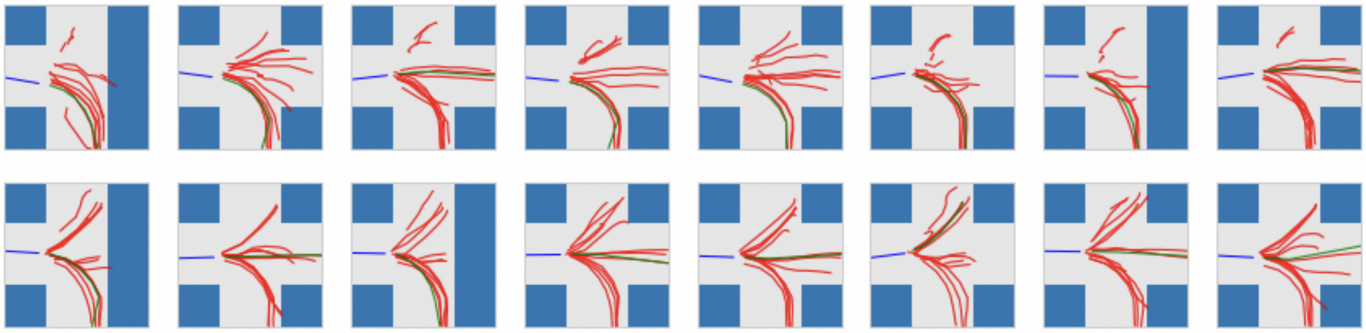


Figure 4.21: **Qualitative results for discovery of left-bound trajectories**. Qualitative results on the two canonical layouts for discovery one-step model with the addition of the cross-attention component (bottom) vs without (top)

The quality of the left going trajectories and the variation among the generated trajectories can arguably be seen as better for the cross-attention model. However, both methods offer different diversity profiles, as highlighted in table 4.3.

As a closing remark, among all the generated trajectories that do not resemble the training distribution, we found an interesting u-turn, depicted in figure 4.22. A rare occurrence that doesn't always show in all model outputs, but interesting nonetheless.

Figure 4.22: **U-turn example**.

4.5 Conclusion

In this chapter, we go further than the previous diversity tasks by tackling the harder problem of discovery. Useful in settings where the data distribution doesn't match the admissibility distribution, but where a data-independent admissibility criterion can be simply defined, discovery can be a useful tool for model robustness. As the field is still in its infancy, not many works have tackled this problem, especially in the context of trajectory forecasting. We bring valuable insight on this topic with three major steps: first, by attempting to squarely define the task at hand, allowing for a clear definition of what can and cannot be done in the context of discovery. Second, by devising experimental setups both for synthetic and real world data, providing a basis for the evaluation of the task as described in step one. Third, we design and test a self-supervised model, to leverage the admissibility information to perform discovery in the absence of training data. Pieced together, these steps constitute a solid basis for further research in discovery, which can also be beneficial for diversity and robustness.

Chapter 5

Conclusion and perspectives

In this closing chapter, we first summarize our contributions, and as the nature of the work done is quite exploratory, we offer perspectives and insight on how the preliminary results obtained during this thesis can be expanded, along with directions that didn't prove fruitful as to provide guidance for future works.

5.1 Summary of contributions

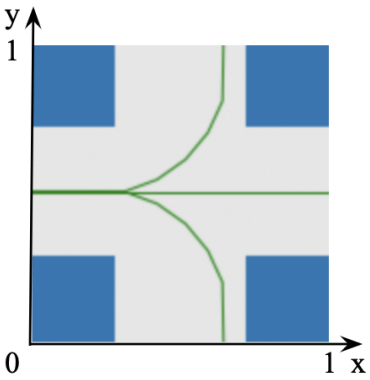
The main topic of interest in this work is predictive diversity. In the context of trajectory forecasting, the nature of real world datasets and the metrics that are associated, like the mean average or final distance error (mADE, mFDE), aim at producing at least one good trajectory among N predicted trajectories. Apart from the best predicted trajectory, very little is made to ensure the $N-1$ other predicted trajectories are meaningful in some way. The structure of generative models used for predicting trajectories, often including sampling from a latent space, is built in a way that promotes sequential sampling of trajectories around a dominant mode with added (often Gaussian) noise. As some works like [Yuan and Kitani, 2020a] investigated diversity on toy problems, we expanded to a real world setting with nuScenes [Caesar et al., 2020] and adapted a method based on determinantal point processes (DPPs) [Macchi, 1975] for the simultaneous generation of a batch of N future trajectories that represent correctly the diversity of the training dataset. As real world datasets pose unique challenges due to being heavily unbalanced towards straight trajectories and having high future predictability given the past trajectory, we introduced several mechanisms to create a diversity-focused trajectory generation method. First, we proposed a DPP kernel, used to compute the similarity between trajectories, that takes into account the angle between all trajectories as well as the distance between them, in order to better promote radial diversity. Second, we propose a layout loss that balances the diversity objective in order to produce trajectories that are admissible under the layout of the scene. Both objectives are carefully integrated in a modular model that can be

adapted on existing trained architectures, in order to provide both good diversity and good accuracy on a representative and challenging real world setting. This work, presented in Chapter 3, led to a publication at ICPR 2022 [Calem et al., 2022].

In addition to this first line of work, we have dug deeper in the question of diversity under the harder angle of data completeness. When explicitly trained for diversity, the previous diversity methods, as well as most generative methods, can only find the kind of diversity that is exhibited in the training data. If we stretch the diversity question further out, we can ask ourselves whether it is possible to generate trajectories that are not represented in the training data at all, but that we nonetheless wish to see appear. In order to tackle this question, we first need to define it, then devise an experimental scheme to test it, and of course find a method that solves our initial problem. In the context of trajectory prediction, the utility of discovery is evident. It enables the incorporation of a structured body of knowledge, which manifests in the data but also exists independently, such as driving regulations or maps of drivable areas that must be respected. By integrating these constraints in a discovery mechanism that can then generate trajectories outside of the training data but admissible under said constraints, we could reduce the reliance on extensive data collection to accurately represent the training distribution necessary for the model to deduce these rules.

The formulation of this problem is as follows. Trajectories $\mathbf{S}$ are represented as a sequence of 2D points. The goal is to predict a set of N possible future trajectories $\hat{\mathbf{S}}_{\mathrm{f}}^{(1:N)}$ given a past trajectory $\mathbf{S}_{\mathrm{p}}$. Given M ground truth modes for possible future trajectories, we have $M-1$ modes in the training dataset. The missing mode has to cover a range of values for either x or y that isn't part of the training dataset, e.g. in the reference frame shown in figure 5.1, missing left trajectories cover the $y \in [0.5; 1.0]$ range whereas training data containing straight and right trajectories cover the $y \in [0.0; 0.5]$ range.

Figure 5.1: **Trajectories Reference Frame**.



Given this setting, the toy environment developed to test discovery allows for precise evaluation of the proposed methods, as we have the several future trajectories as ground truth that we can use to assess whether the model discovered a new trajectory or not.

The first challenge lies in finding a model that balances two seemingly opposite objectives. First, the model has to be stable enough with respect to the training distribution such that the generated elements still exhibit trajectory-like patterns. Second, the model has to be flexible enough to produce trajectories that go further from what has been seen in the training data. The architecture and training detailed in Chapter 4 provide a proof of concept model that validates the possibility of generating new

trajectories that progressively drift away from the training distribution to reach the missing modality. The model is based on an encoder-decoder framework, which enables the creation of a latent space in which we can control the diversity. The decoder is designed such that its only input come from the latent space, allowing for a greater control on diversity as the latent code has to contain enough information to generate the future trajectory.

5.2 Autonomous discovery perspectives

5.2.1 Selection functions

The model laid out in Chapter 4 to generate diversity provides an interesting proof that generating a controlled drift from the training distribution to a broader distribution that is still admissible is possible. However, a significant limitation is that it uses the ground truth for the selection function of which generated trajectory to reintegrate in the self-labelling training scheme. The proposed model is still an advancement towards fully autonomous discovery, as it doesn't use the ground truth as a supervision for the generated trajectories, but in order to be usable in settings other than toy datasets, it should be self-reliant for the selection function.

In order to provide insight for future work towards this goal, we detail here strategies that have been implemented for the selection function and didn't prove to be effective.

Output space distance selection The most intuitive way of finding the most interesting predicted trajectories that are to be selected and included in the training dataset as self-labeled trajectories that expand the training distribution is to use heuristics in the output space. Several strategies have been tested in this space:

- **Mean Outputs.** As training progresses, we can keep an average of predicted trajectories. At any timestep, when we generate a set of N possible future trajectories, we can compute their distance with respect to this mean past generated trajectory, and compare each new generated trajectory to this (moving) average. Generated trajectories that are furthest to this average trajectory can be picked as potentially covering new ground. However, it resulted in degenerate solutions that picked trajectories that weren't valid and poisoned the training dataset.
- **Mean GT.** During training, we can gather the ground truth seen at each step instead of previous outputs, and compute a mean trajectory, indicative of what has been seen before. By picking the furthest one to integrate back in the training dataset, the intuition is that trajectories that are far from seen ground truth have more chance to be completely new. This solution was very

similar to the previous Mean Outputs solution, and often picked invalid trajectories that were stationary.

- **Mean GT (warmup).** In order to mitigate this issue of picking stationary trajectories, the same trajectory was employed for selecting promising trajectories, but after some initial warm-up period of training that is devoted to letting the decoder stabilize so the generated trajectories were not stationary or otherwise invalid. This strategy successfully improved the quality of the trajectories selected for inclusion in the self-labeling scheme, but didn't succeed in selecting trajectories different enough to expand the generated distribution and achieve discovery in any meaningful way.

- **Mean GT (endpoints only).** As an effort to mitigate the issue causing a degenerate trajectory to be picked for reintegration, a tentative simpler method was used, using only the endpoints to each trajectory to compute the distance to the mean trajectory. The stability of the resulting method wasn't satisfactory as most models diverged during training.

- **Zero point.** As a sanity check on the Mean GT endpoints method, an experiment has been run using the zero point as an anchor point to compare the endpoints of predicted trajectories, in order to select the trajectory with the endpoint furthest from the (0,0) point. Using this fixed point proved to be much more stable, with no diverging training, and even some runs that reached a recall above 0.5, indicating a successful discovery of left trajectories (see table 5.1 and figure 5.3). However, it didn't prove reliable enough to generate a full distribution including the left-going trajectory without generating degenerate trajectories in the process.

Latent space distance selection A selection function in the output space is intuitively motivated, however much more subordinated to a hand crafted choice of distance and thus looses generality as a method. In order to explore other avenues that would make the method more generic, we also investigated the possibility of selecting the trajectory to be reintegrated in the training dataset on the basis of its latent representation. The goal is to find a way to identify a latent that is further from the others yet still in an admissibility region. The first experiment we have tried in this regard was simply to pick the trajectory with the most distant latent code z. However, this didn't prove to be an effective method, as evidenced by the fact that performance quickly degraded as the addition of unsuitable trajectories poisoned the dataset.

Even if this first attempt didn't work, it is notable to mention that it is most likely due to the lack of proper organization of the latent space induced by the training method we use (using the reconstruction objective from the VAE without the KL regularization).

As such, we have promising future directions for devising a selection function based on first

constraining the latent space to exhibit some kind of arrangement so that we can reason within it and find suitable exploration direction directly within the latent space. VAEs, and any model that exhibits the property of modeling a latent space during training could be used to propel discovery. But it is necessary, for such latent space discovery, to have a minimum of organization in the latent space. In [Notin et al., 2021], the authors note that "the lower-dimensional continuous representation of objects allows to transform the original discrete optimization problem to a simpler continuous optimization one in the latent space". This sentence might seem generic, but it embodies the idea, exhibited in other papers, that working in the latent space might be an easier way to solve for certain problems than working in the output space. Their task is to generate a sample that must satisfy an expensive admissibility function to be valid. In order to take into account this admissibility in the sampling, they used decoder uncertainty as a proxy for admissibility and managed to avoid regions in the latent space with high uncertainty, as they are likely to produce invalid outputs. While our problem is quite opposite, as our admissibility function and uncertainty from training data are not the same, this work illustrates that some methods can be used to optimize sampling in the latent space to satisfy external constraints.

Following the reasoning that working in the latent space can yield interesting results in the field of diversity, [Chadebec and Allassonnière, 2022] provide an interesting perspective on the training of VAEs as seen through the lens of geometric learning. In essence, they argue that the latent space created during the training of a VAE can be viewed as a Riemannian manifold, and consequently we can use Riemannian geometry tools to navigate on it. For sampling, instead of traditionally using an Euclidean Gaussian, they provide tools for sampling along the equivalent version but on the manifold, following geodesics instead of straight Euclidean lines. This work is of particular interest to diversity and discovery as we could potentially further these tools to not only sample along the geodesics, but also determine the directions depending on the potential diversity of each direction.

Those works are interesting pointers towards one direction: it is probably possible to derive a learning scheme using some form of manifold learning or geometric learning [Bronstein et al., 2017]. The basis of such a work would be to incorporate regularization so that the latent space correctly represents regions associated to different modalities, then leverage the admissibility function to guide sampling on the latent space based on the possible diversity we can generate from one sample. If in [Chadebec and Allassonnière, 2022], applying the equivalent of a Gaussian sampling in the latent space for sampling yields results that are clearly separated for each class, we can imagine that weighting this sampling distribution according to the discovery potential of each direction could possibly provide interesting results for both diversity and discovery.

Cross-Attention weights selection The first working model for discovery, named "Oracle SL" in Table 4.2, didn't include any attention mechanisms. The addition of it, detailed in section 4.3.4, eventually improves both the qualitative results and the distance between the predicted trajectories to the ground truth left trajectory we want to discover, so we kept the mechanism in the final model architecture. It is, however, not an essential part of the model as a recall of 0.97, indicative of successful discovery, is reached with the Oracle SL version of the model.

This addition was, originally, part of an experimentation to find a suitable self-selection function for the best trajectory to integrate in the self-labeling mechanism. The attention weights $\mathbf{A} = \sigma(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}})$ are computed before multiplying with the original value matrix $\mathbf{V}$. This matrix $\mathbf{A}$ can be seen as a contextual similarity matrix relating the latent codes with the spatial features of the layout. The intuition for this experimentation was to leverage this similarity to spot generated latent codes that stand out from the average attention exhibited by previous latent codes, so that latent codes attending to different spatial features would be drifting from the past distribution. The details for the computation of matrix $\mathbf{A}$ are laid out in Figure 5.2.

As such, the initial goal of this architectural addition was to leverage the attention weights in order to perform the selection of the best trajectory. Here are the main steps:

1. The attention weights $\mathbf{A} \in \mathbb{R}^{BS \times N \times HW}$ are computed for each batch. N is the number of predicted trajectories and HW the dimensionality of spatial layout embeddings l_{spatial} of the drivable area layout.

2. We aggregate these weights over time and trajectories to obtain an average attention map, that should represent the relationship between space and previously predicted trajectories.

3. For each new batch of generated trajectories, we also have the associated attention weights, that we can compare with the aggregated weights from previous trajectories.

4. The generated trajectories whose attention weights are furthest from aggregated weights can be considered as most novel, at least in the attention with the spatial map aspect, and selected for integration in the training step of the next epoch.

So far, the experimentation remains unsuccessful (see Table 5.1), with a model that doesn't manage to pick the best trajectory to achieve discovery. However, the experiments undertaken so far using this reasoning were not extensive, and might constitute a solid basis for finding an autonomous selection function mechanism. Adding patches, moving averages, and understanding better the representation made by the attention weights, integration with the admissibility function, all could converge to provide a working autonomous discovery selection function.

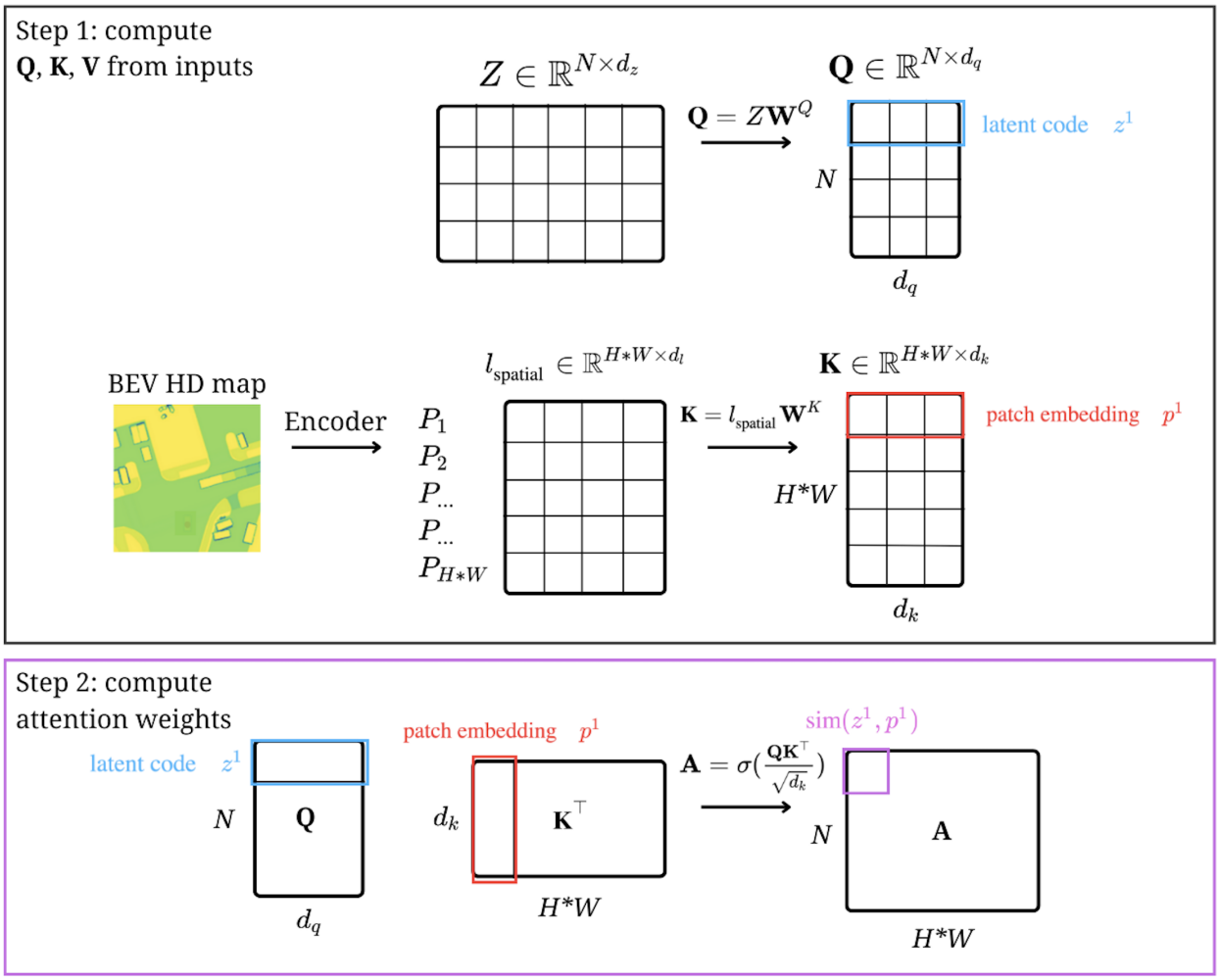


Figure 5.2: **Cross-Attention weights computation**. Detailed explanation of the elements involved in attention weights $\mathbf{A}$ computation: first, the query matrix $\mathbf{Q}$ is computed from the latent codes matrix $Z \in \mathbb{R}^{N \times d_z}$, representing the set of N latent codes $z^i, i \in \{1, \cdots, N\}$, each of which is a vector of dimension d_z. Each latent code z^i is to be decoded via the decoder to produce future trajectories. Z is used to produce the keys while the Bird-Eye-View HD map is used to produce a spatial encoding l_{spatial} of size $(H \times W \times d_l)$ (we use spatial dimension $H = W = 14$ and flatten them so $l_{\text{spatial}} \in \mathbb{R}^{HW \times d_l}$) which is used to produce matrices $\mathbf{K}$ and $\mathbf{V}$. The second step is to compute matrix $\mathbf{A}$ of attention weights with $\mathbf{A} = \sigma(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}})$, where σ is the softmax function. The resulting cross attention weights $\mathbf{A}$ are used as a contextual similarity matrix between the latent codes and the spatial embeddings that could guide the selection of promising z^i for controlled drift of the training distribution towards spatially different codes that can be decoded in novel trajectories.

Quantitative results The quantitative results of these strategies are shown in table 5.1. The *Best* and $Best_{10}$ metrics show the average distance (in meters) over all trajectory points between the missing left ground truth trajectory and the closest trajectories generated by the model. High average distance

means the closest trajectory is quite far from the ground truth missing trajectory, while lower values of 1m or lower is generally indicative of good performance. The recall, computed over the two trajectories we want to find for each sample (the actual future ground truth of the past trajectory and the left missing trajectory), conveys three major indications: a recall close to 0.0 means the selection function introduced data that weren't representative of trajectories, corrupting the dataset and making the model diverge, unable to even generate data from a modality seen in the original training dataset. A recall around 0.5 is indicative of a stable model that has not been able to discover trajectories close enough to the missing ground truth, and a recall close to 1.0 means discovery success. For reference, the first line (Baseline) represents the model trained without any self-labeling, where there is no incentive towards shifting the output distribution. The "CA Oracle" model refers to the performance of the best model detailed in Chapter 4.

Selection	Best	Best_{10}	Recall	mADE	mFDE	rF	DAC	ASD	FSD
Baseline	3.96	4.64	0.50	1.559	3.060	2.399	0.993	3.287	4.705
CA Oracle	0.70	0.72	0.97	0.697	1.324	5.214	0.938	3.301	5.689
CA	2.834	3.587	0.488	1.884	3.087	4.101	0.733	13.902	15.040
Mean GT †	4.12	4.67	0.0	-	-	-	-	-	-
Mean GT + warmup †	3.214	3.329	0.498	-	-	-	-	-	-
Mean GT endpoints †	-	-	-	-	-	-	-	-	-
Mean outputs	2.748	3.089	0.501	1.399	2.337	3.352	0.921	3.489	5.529
Zero point	2.258	2.637	0.506	1.400	1.769	5.439	0.904	4.902	6.851
Mean z †	3.213	3.814	0.167	-	-	-	-	-	-

Table 5.1: **Selection methods discovery results**. Discovery and diversity metrics for the selection function presented in this section. The results presented are averaged on 3 random runs. The recall metric in particular indicates whether the left-going trajectory has been found, with values close to 0.0 indicating that no trajectories have been found (in the case of a diverging model), 0.5 that only one of the two correct trajectories have been found (typically the right of straight-going ground truth trajectory), and a perfect score of 1.0 would mean that for each past trajectory, both the actual ground truth and the left-going trajectories are generated.
† These models diverge and produce degenerate solutions as the training dataset becomes polluted. We report discovery and diversity metrics at the last epoch before divergence.

It is interesting to note that in some experiments looking into the effectiveness of using the mean outputs and the zero point as a selection function comparison point, the recall went slightly above 0.5 (the best experiment among all launched went to 0.522). The results were encouraging enough to find the left trajectory some of the time, as opposed to never in all the other self-contained methods, but not reliably enough to provide a meaningful avenue for performance. The visual results these experiments were convincingly finding the left trajectory some of the time (see figure 5.3), but too many artifacts and degenerate trajectories are generated in the process.

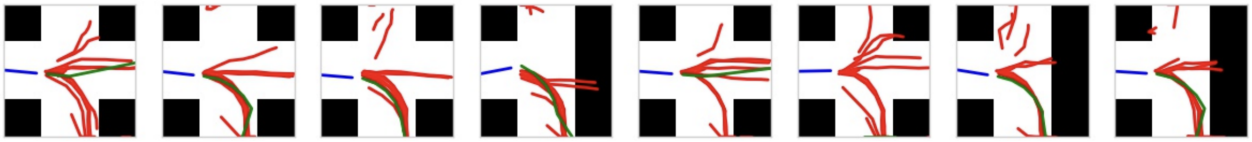


Figure 5.3: **Left-going trajectories with an independent selection function**. Trajectories generated by the "zero-point" model, exhibiting the capability of finding left trajectories but generating many degenerate trajectories in the process.

5.2.2 Real-world trajectories

Evaluating discovery in a real-world driving dataset, while an important step in the usability of the research in more industrial settings, remains a challenging task for both diversity and, by extension, discovery. The main challenge of a real-world setting is that the data distribution of training examples is unimodal, i.e. only one future trajectory exists for one past trajectory. It is a natural step for discovery but requires a fully autonomous system, including a selection function independent from the missing modality ground truth. Nevertheless, ground work has been done for future usage of real-world datasets for discovery.

The first step to make nuScenes usable for discovery in a setting comparable to the toy dataset (i.e. one modality is removed from the training data) is to determine which direction each trajectory go, in order to correctly select a subset of the dataset, omitting the left modality. In order to find the orientation of each trajectory, we compute the curvature κ of the available 12852 total trajectories in nuScenes.

As nuScenes trajectories are polylines composed of discrete 2D vertices, we use the following polyline curvature approximation formula, which works when vertices are evenly sampled (2Hz in the case of nuScenes):

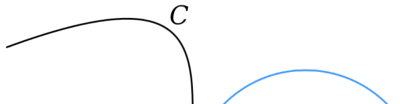
$$\kappa = \frac{2 * ((x_2 - x_1) * (y_3 - y_2) - (y_2 - y_1) * (x_3 - x_2))}{\sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2) * ((x_3 - x_2)^2 + (y_3 - y_2)^2) * ((x_1 - x_3)^2 + (y_1 - y_3)^2)}}. \tag{5.1}$$

This formula provides an estimation of the signed curvature of a planar curve at point (x_1, y_1) using two neighboring points $(x_2, y_2), (x_3, y_3)$, which we compute for all the trajectory points (save for the end points) then average over the whole trajectory.

To give some context over where the formula comes from, the curvature κ of a planar curve at point P was historically defined as the inverse of the radius of the osculating circle best approximating the curve at P (Figure 5.4).

Although polylines don't really have a curvature per se (they are composed of straight line segments

Figure 5.4: **Osculating circle**. Historically used to compute the curvature of a planar curve, the osculating circle of radius r best approximates the curve at one point P. The curvature at this point is then $\kappa = \frac{1}{r}$.



which have 0 curvature), we can approximate a curvature at each vertex looking at neighboring vertices. Considering a triangle ABC on this osculating circle, with side lengths a, b and c, from the law of sines the diameter can be expressed as $\frac{a}{\sin A} = \frac{b}{\sin B} = \frac{c}{\sin C} = 2r$. The area A of the triangle being $A = \frac{1}{2}ab\sin C$, substituting $\sin C$ we get $r = \frac{abc}{4A}$. As $\kappa = \frac{1}{r}$, we get equation 5.1 with the numerator being $4A$ and the denominator being the product of the sides lengths.

Using this formula, we get a good approximation of the curvatures κ of the polylines corresponding to the future trajectories in the dataset. Table 5.2 shows the distribution of the curvatures for the nuScenes dataset.

Total	$\kappa < -0.5$	$-0.5 < \kappa < -0.01$	$-0.01 < \kappa < +0.01$	$+0.01 < \kappa < +0.5$	$\kappa > +0.5$
12694	205	1896	8565	1795	233

Table 5.2: **Curvatures of nuScenes trajectories**. Out of the 12853 nuScenes trajectories, 12694 have a computable curvature κ. Among these, 12256 are non-outlier trajectories in the $[-0.5; +0.5]$ range and can be classified as going left ($-0.5 < \kappa < -0.01$), straight ($-0.01 < \kappa < +0.01$), and right ($+0.01 < \kappa < +0.5$).

Given that nuScenes data have some stationary trajectories incompatible with curvature computation, we only consider trajectories in the $[-0.5; +0.5]$ range for κ, as they correspond to usable trajectories. The range $[-0.01; +0.01]$ corresponds to straight trajectories, and the remaining bins $[-0.5; -0.01]$ and $[-0.01; +0.5]$ correspond to left and right trajectories respectively, with most (96%) trajectories being in the $[-0.1; +0.1]$ range.

As expected, the dataset is heavily unbalanced with 69.88% of non-outlier trajectories being straight trajectories. Left and right-turning trajectories are however balanced, with 15.47% and 14.65% for left and right modalities respectively. Figure 5.5 shows examples of both left and right trajectories as referenced by the polyline computation of equation 5.1.

In nuScenes, some trajectories are stationary, both in the past and in the future. For these, GPS-uncertainty renders the curvature computation impossible. As trajectory points are closer, the curvature computation of κ produces a higher value, up to infinity in the case of overlapping points of a stationary trajectory. As they don't provide any value to training for trajectory forecasting, diversity and discovery tasks, we discard them. Figure 5.6 provides an illustration of such trajectories.

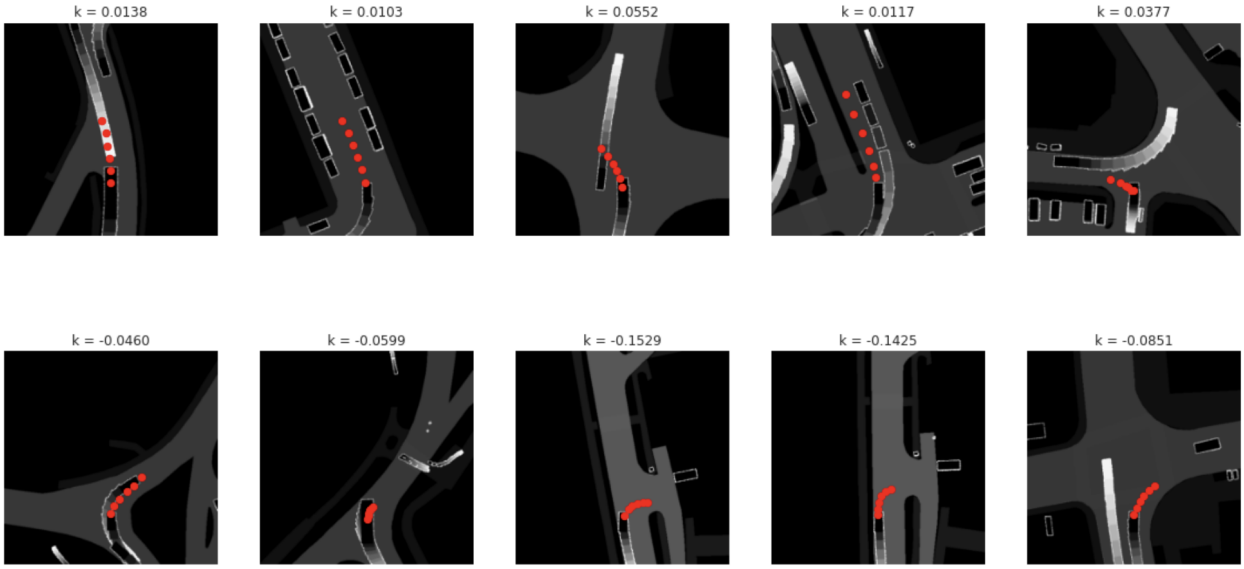


Figure 5.5: **Examples of nuScenes trajectories by curvature**. Plot of futures trajectories (red dots) on top of bird-eye-view HD maps of the layout. (top) negative curvature (i.e. going left) trajectory examples. (bottom) positive curvature (i.e. going right) trajectory examples. Higher absolute values don't correlate with higher curvature but rather how far apart the points are, but we can see that these trajectories all go in the direction indicated by the curvature measure, validating the formula.

Figure 5.6: **Examples of nuScenes stationary trajectories**. Stationary trajectories, numerous in nuScenes, are discarded during training as they don't provide much value and can skew the evaluation metrics, especially accuracy as they are "easy examples". For discovery, we also discard them due to their irrelevance to the task at hand.

5.3 Image discovery

Discovery in the context of trajectory forecasting is a setting that allows for the integration of well defined constraints such as drivable area map. During this thesis work, a brief exploration outside the realm of trajectory forecasting was allowed to assess the possibility of applying discovery methods to image generation.

The main challenge with image discovery lies in determining the admissibility function. The

circular trap of deriving an admissibility function that describes the desired outcome so precisely that the whole generation process isn't useful anymore is the most difficult problem of out-of-distribution generation. In trajectory prediction, it is easier to avoid it since a lot of external constraints are available, like driving rules or driving area. The delicate part of out-of-distribution generation lies in balancing the generative process that has to be data-dependent enough to output elements that have the same identity as elements in the training dataset, like trajectories or shapes, but that are free enough from this same training distribution to represent unknown variations of these elements, without completely altering the identity part. This dichotomy is reflected in the architecture of discovery models as the balance between choices made to generate varied outputs, and admissibility function to constrain the diversity. In image discovery like other tasks, the discovery process is split into two parts, first generating samples different enough from the training distribution to allow for exploration, second including an admissibility function that has to constrain the generated samples. Although interesting steps were taken in this thesis to tackle the first problem, the admissibility function wasn't investigated. The present section presents the preliminary steps towards generation of out-of-distribution images.

5.3.1 dSprites exploration

[Montero et al., 2020] and [Montero et al., 2022], mentioned in Chapter 4, explored the relationship between disentanglement capability of models and their combinatorial generalization power using image generation datasets as an experimental setting. The dSprites dataset [Matthey et al., 2017], used in that work, is a dataset of shapes that are determined by a combination of five generative factors: shape (among heart, square and ellipse), scale, rotation, x-position and y-position. While useful to study disentangling effects on generation, this type of dataset is also useful to study exploration. The exploration task in the nomenclature laid in [Montero et al., 2020], detailed in 2.4.2, used dSprites to derive an exploration task in which every image containing a shape with x-position > 0.5 (which is all shapes on the right side of the image) were removed from the dataset. While the model used to test for combinatorial generalization and exploration in [Montero et al., 2020] and [Montero et al., 2022] allows for an explicit querying of the desired generative factors, the "organic" exploration in the discovery task has to be intrinsic to have any value, because knowing in advance what we want would defeat the purpose. Instead, we want to rely on an admissibility function that is only able to describe if the generated example is admissible or not.

In order to test discovery in the image setting, a quick exploratory work has been made on dSprites to test the first step of discovery: generating samples that deviate from the training distribution. A relaxed VAE of the same architecture presented in 4.3.5 was trained on images from dSprites dataset, all images containing shapes on the right side of the image (x-position > 0.5) having been removed. For trajectory forecasting, the diversity lies in the difference in the generated future trajectories, that

follow the input past trajectory. It is harder to devise such an intuitive past / future separation for images. The setting we chose is as follows: the input is one image from the dataset, and the model generates a set of N latent codes z that are decoded into N images. The goal is to generate images that retain one generative factor from the original image but are different on every other aspect. In order to do that the first step is to generate images that differ from the original image in ways that we can control to expand the generated distribution to data that have not been seen during training, in this case shapes on the right of the image.

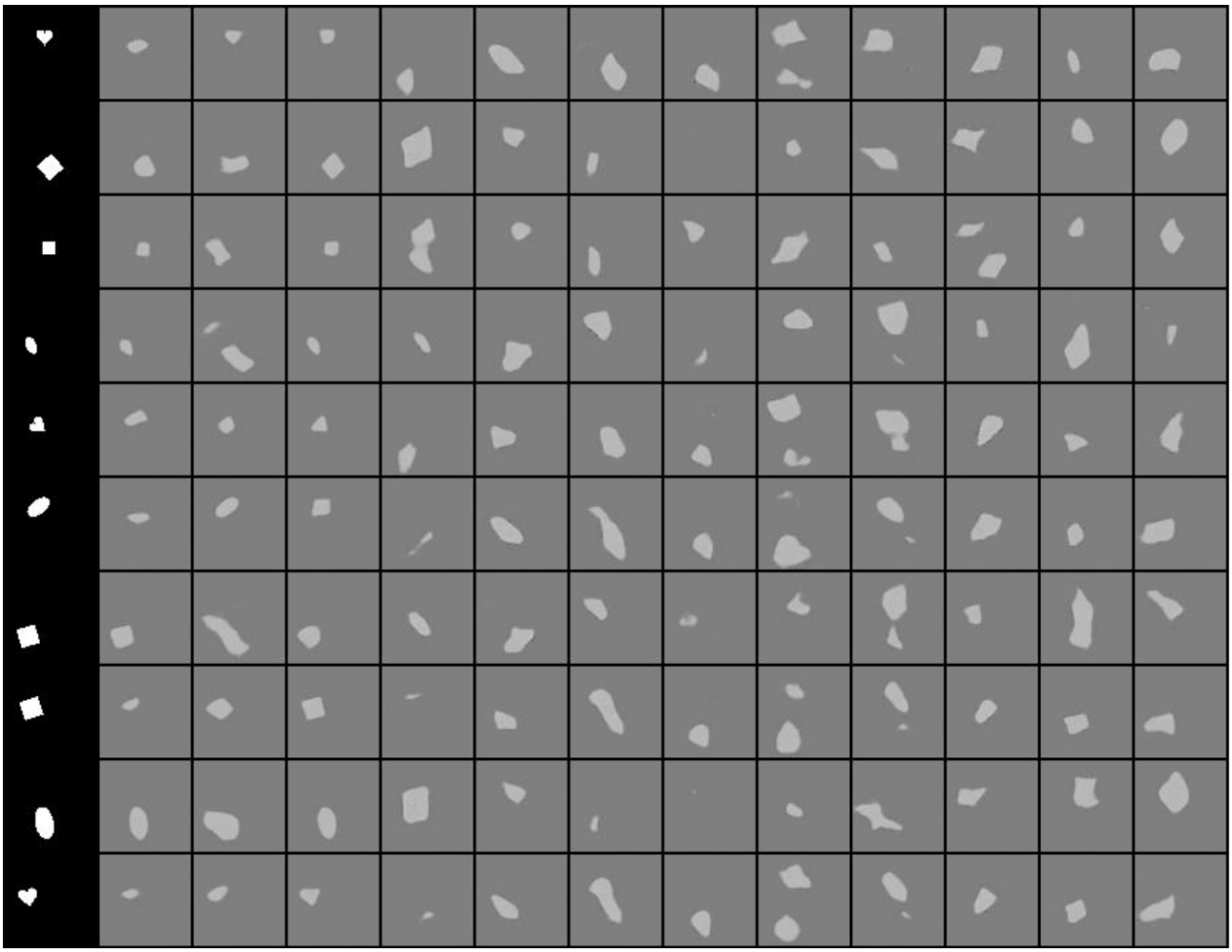


Figure 5.7: **Shapes generated from dSprites through a relaxed VAE**. Each row represents an example. For each example, the left column (high contrast) represents the input image, from which $N = 12$ images are generated (left columns in the same row). The model is trained to generate images that are different from the input image while retaining one of the generative factors of the input image as an admissibility function. The goal is to generate shapes in the right part of the image, that have been unseen during training.

The relaxed model we used managed to generate shapes that were different from the original

image (see figure 5.7 for examples), but the relaxation meant that we had insufficient control on the generated shape to then coerce the model to generate the original shapes in a different position.

5.3.2 Diffusion models

Image generation is currently dominated by diffusion models. However, their powerful generative power is built on datasets of considerable size. Denoising Diffusion Probabilistic Models (DDPMs) [Ho et al., 2020] are the the basis of many successful models, and could be used for discovery in the context of images.

As a quick primer on diffusion models aimed at giving the minimum context needed to understand the preliminary work conducted on this subject, diffusion models are based on the eponymous process known as diffusion. This process involves gradually adding noise to image data over time and learning to reverse this process, allowing the trained model to generate or reconstruct data by systematically reducing the noise. The learning process typically occurs in two phases: the forward (noising) phase and the reverse (denoising) phase:

Forward (noising) Phase In this phase, a data point (like an image) is gradually corrupted by adding noise over several steps. The process can be described by a Markov chain, where the state at each time step t is increasingly noisier than the previous step. This process can be explained by the simplified equation $X_{t+1} = X_t + \beta_t \epsilon_t$, where X_t is the data at time step t, β_t is a noise scale factor, and ϵ_t is the noise term.

Reverse (denoising) Phase In the reverse phase, the model learns to reverse the noising process. Starting from pure noise, it gradually reconstructs the data by learning to predict the noise that was added at each step in the forward phase and then subtracting it. This process involves training a deep learning model to estimate the reverse of the diffusion process.

During this thesis, a small exploration work has been done in order to assess the feasibility of discovery for images using diffusion models. The base equation $X_{t+1} = X_t + \beta_t \epsilon_t$ from which the noised images are generated is mirrored in the denoising phase, where the model removes a portion of the noise at each step. For every step, a gradient is computed and the image moves to the next along the computed gradient. To allow for exploration, during the early denoising phase one could "overshoot" the gradient by moving the image along the direction of the gradient but further than originally computed, to better explore the noise space and potentially find new noised images in the direction that could be subsequently denoised in more diverse denoised images. Figure 5.8, illustrates the reconstructed images from celeb-A by overshooting the gradient.

Figure 5.8: **Images generated through diffusion by overshooting the gradient**. (top) by moving the gradient a little. (bottom) a lot. The discrepancy illustrates that we move in the noise space in a way that isn't structured enough to meaningfully alter the faces of the generated subjects.

Unfortunately, the faces prove difficult to reconstruct after overshooting the gradient in the noise space. Exploration in the noise space doesn't seem to prove neither practical or powerful, but this kind of exploration could be done in the latent space of diffusion models that include one, such as [Rombach et al., 2022].

5.4 Closing remarks

From the broad title of "action and trajectory prediction for autonomous driving", a specific aspect of trajectory forecasting was explored: diversity. Making use of an elegant mathematical structure, determinantal point process, advances have been made to improve diversity in the context of trajectory forecasting, reliable even in real world data settings. Furthering the diversity question, which relies on an admissibility function external to the dataset to constrain the diversity promoting effects, we asked ourselves whether this admissibility function could be used to guide the exploration of new samples outside of the training distribution. As a more exploratory work, a toy dataset was first derived in order to provide a clear experimental setup able to assess what "discovery" means and how to consider that what is missing has been successfully discovered. Building on that testbed, a method has been proposed to carefully loosen the generation process, enough to go out of the training distribution but controlled enough so that generated trajectories still resemble trajectories. Further research is needed to have a fully autonomous discovery system, but the groundwork has been laid and a first model has been proposed to validate the possibility of discovery, as our generative model is able to gradually move away from the training distribution towards a broader distribution of admissible trajectories, taking back in training only trajectories that have been previously generated.

As the work is constrained by both time and industrial purpose, diversity and discovery have mainly been explored in the context of autonomous driving trajectory forecasting. However, the techniques and explorations developed in this work can be expanded to all settings that follow a similar pattern: first, some kind of encoder-decoder method has to be employed as to form a latent space from which to sample. Second, an admissibility measure has to been provided to constrain diversity and guide discovery. This second condition is where one needs to be meticulous, in order to avoid the trap of building an extensive admissibility function that is sufficient to describe the distribution of what we want to generate. The admissibility criterion has to be independent from the training data, easy to understand and evaluate but not comprehensive enough so one could merely create a diffusion generator with this admissibility function for training so that a perfect distribution can be crafted from noise using only the admissibility function. The discovery setup has the training dataset laying out the "identity" of the elements that need to be sampled (i.e. the trajectories have to resemble trajectories) and the admissibility has to provide guidance on which areas of the latent space can be expanded.

The first work described in this thesis provides an interesting way of looking at the broader issue of generative models, and can open up a perspective to increase the diversity of outputs from generative models and maybe a way to improve the intrinsic discovery of elements not present in the training dataset, either to reduce the need of enormous datasets, or to improve the ability of such models to generate novel things.

Résumé de la thèse

Introduction

Dans le domaine de la conduite autonome, l'anticipation précise des trajectoires des véhicules et des piétons est cruciale pour garantir la sécurité sur la route. Cette thèse, intitulée "Prédiction d'actions et de trajectoires pour la conduite autonome", se concentre sur l'amélioration des modèles prédictifs en introduisant des mécanismes de promotion de la diversité et de découverte dans les prédictions de trajectoires. Ces mécanismes permettent aux véhicules autonomes de mieux gérer la complexité et l'incertitude des environnements routiers en prévoyant non seulement la trajectoire la plus probable, mais aussi un ensemble de trajectoires possibles et réalistes, y compris celles qui pourraient être critiques pour la sécurité.

Les systèmes actuels de conduite autonome reposent généralement sur des modèles de prédiction qui sont formés à partir de données historiques de trajectoires. Ces modèles, souvent basés sur des autoencodeurs variationnels (VAEs), ont pour objectif de prédire la trajectoire future la plus probable en se basant sur les trajectoires passées et l'environnement immédiat. Cependant, cette approche peut être limitée dans les situations complexes où plusieurs trajectoires futures sont possibles. Les systèmes pourraient ainsi manquer de flexibilité et échouer à anticiper des événements rares mais critiques, augmentant le risque d'accidents.

La problématique centrale de la thèse, qui interroge sur la manière de concevoir des modèles capables de générer une diversité de trajectoires futures tout en maintenant un haut niveau de précision, ainsi que sur la garantie que ces trajectoires incluent des scénarios rares mais potentiellement dangereux souvent sous-représentés dans les données d'entraînement, découle de la nécessité de renforcer la sécurité et la robustesse des systèmes de conduite autonome. Les modèles actuels se concentrent principalement sur la prédiction de la trajectoire la plus probable, négligeant ainsi une multitude de scénarios possibles qui, bien que rares, peuvent être critiques pour éviter des accidents. Cette approche traditionnelle limite la capacité des systèmes à anticiper des situations complexes et imprévisibles, telles que des manœuvres brusques ou des changements inattendus de comportement des autres usagers de la route. Pour répondre à cette problématique, l'investigation de la diversité dans les prédictions de

trajectoires est apparue comme une solution prometteuse, car elle permet de couvrir un éventail plus large de futurs possibles, offrant ainsi une meilleure préparation à des situations variées et inattendues. En parallèle, l'exploration du concept de découverte s'est avérée essentielle pour pallier les limites des données d'entraînement en identifiant et en générant des trajectoires qui ne sont pas présentes dans les ensembles de données, mais qui sont admissibles et pertinentes pour la sécurité. Ensemble, ces deux axes de recherche – la diversité et la découverte – permettent non seulement d'élargir l'horizon des prédictions, mais aussi d'intégrer des scénarios critiques non observés durant l'entraînement, contribuant ainsi à la conception de systèmes de conduite autonome plus fiables et plus sûrs.

Nous avons donc développé dans cette thèse les méthodes de diversification des trajectoires générées, en suivant deux approches majeures :

1. **Promotion de la diversité dans les prédictions de trajectoires** : Développer des mécanismes qui permettent de générer non seulement la trajectoire la plus probable, mais aussi un ensemble diversifié de trajectoires futures. Cette diversité est essentielle pour la robustesse du système face à des scénarios variés et imprévus.

2. **Découverte de nouvelles trajectoires** : Introduire des méthodes pour découvrir des trajectoires qui ne sont pas représentées dans les données d'entraînement, mais qui sont néanmoins admissibles (i.e. possibles par rapport à l'environnement de conduite), et pertinentes sur le plan de la sécurité, car la connaissance de toutes les trajectoires futures possibles permet d'améliorer la précision des prédictions. Ces trajectoires doivent être admissibles selon des critères externes définis par des fonctions d'admissibilité.

Génération d'exemples dans les modes minoritaires

La diversité dans les modèles génératifs, un aspect crucial pour la prédiction de trajectoires en conduite autonome. L'objectif est de surmonter les limitations des modèles génératifs classiques qui tendent à se concentrer sur les modes majoritaires, négligeant ainsi les scénarios rares mais potentiellement dangereux. En effet, la capacité à prédire non seulement la trajectoire la plus probable, mais aussi un ensemble diversifié de trajectoires futures, est essentielle pour garantir la sécurité des véhicules autonomes, en particulier dans des environnements complexes et imprévisibles. Ce premier travail propose une méthode novatrice pour prédire un ensemble structuré de trajectoires diverses, en complétant un modèle génératif sous-jacent par un composant de diversité basé sur un processus ponctuel déterminantal (DPP).

La prévision des trajectoires futures pour les utilisateurs de la route est une tâche intrinsèquement stochastique et multimodale. À tout moment, plusieurs actions de conduite admissibles mais très différentes peuvent être prises par chaque agent, en particulier aux intersections ou sur des routes à plusieurs voies. Ignorer certaines de ces trajectoires possibles peut entraver le système de conduite autonome, car la planification des mouvements du véhicule dépend fortement de la précision et de la diversité des prédictions de trajectoires. Les modèles traditionnels, bien que capables de prévoir plusieurs sorties, manquent souvent de diversité en raison de leur formation sur des ensembles de données contenant principalement une seule trajectoire future par scénario.

Le passage à une prédiction stochastique des trajectoires futures, en utilisant des modèles génératifs, a marqué un tournant dans ce domaine. Cependant, la distribution des sorties produites par ces modèles est souvent trop centrée sur les modes majoritaires des données d'entraînement, telles que les trajectoires en ligne droite. Cela est particulièrement problématique dans les ensembles de données réels où les scénarios de virages ou de manœuvres complexes sont sous-représentés. Par conséquent, de nombreux modèles génératifs, comme les autoencodeurs variationnels conditionnels (cVAEs), souffrent de ce qu'on appelle un effondrement des modes, où seules les trajectoires les plus probables sont échantillonnées, laissant de côté des trajectoires moins probables mais cruciales.

DIVA La prédiction probabiliste de trajectoires repose sur des modèles génératifs composés d'un encodeur, qui projette les données d'entrée à haute dimension vers des représentations latentes de plus faible dimension, et d'un décodeur, qui génère les trajectoires futures à partir de ces représentations. Le problème clé abordé dans ce travail est la difficulté de garantir une diversité structurelle des trajectoires prédictes, tout en maintenant leur admissibilité dans la zone de conduite.

Le modèle DIVA, introduit dans cette thèse, est un modèle de prévision multi-sorties conçu pour prédire des trajectoires diversifiées mais admissibles. La méthode se base sur un modèle classique de prédiction de trajectoire : le VAE.

Les autoencodeurs variationnels sont des modèles génératifs puissants qui permettent de générer des trajectoires futures en apprenant une distribution latente à partir des données d'entraînement. Le VAE utilise une fonction de loss qui combine la reconstruction des données d'entrée et la régularisation de la distribution latente. La fonction de loss typique pour un VAE est donnée par :

$$\mathcal{L}_{VAE}(\theta, \phi) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \mathrm{KL}(q_\phi(z|x)||p(z))$$

Où : $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$ est le terme de reconstruction qui pousse le décodeur à générer des données similaires à celles d'entrée. $\mathrm{KL}(q_\phi(z|x)||p(z))$ est le terme de régularisation qui force la distribution latente $q_\phi(z|x)$ à rester proche d'une distribution a priori $p(z)$ (souvent une distribution normale

standard).

Cependant, les VAEs standard tendent à produire des trajectoires concentrées autour des modes majoritaires de la distribution d'entraînement, ce qui limite la diversité des trajectoires générées.

Pour surmonter les limitations des VAEs en termes de diversité, DIVA introduit deux mécanismes :

- le remplacement de la sélection séquentielle des codes latents z par une Diversity Sampling Function (DSF) apprise a posteriori du modèle génératif sous jacent et chargée de produire N codes latents z, N étant le nombre désiré d'exempes générés en sortie du décodeur.
- l'utilisation des Determinantal Point Processes (DPPs) en tant que fonction de coût pour apprendre la DSF.

Diversity Samping Function. L'architecture DSF repose sur une approche en deux branches distinctes, qui travaillent en synergie pour garantir la diversité des trajectoires générées tout en maintenant leur qualité et leur réalisme :

1. **Branche de diversité.** Cette branche est responsable de la promotion de la diversité dans les trajectoires générées. Elle prend en entrée la représentation encodée de la trajectoire passée (notée h) et produit plusieurs codes latents partiels $z_p^{(n)}$. Ces codes sont conçus pour capturer des variations dans les trajectoires futures potentielles en mettant l'accent sur différents aspects du mouvement passé.
2. **Branche de qualité.** La branche de qualité, quant à elle, se concentre sur le maintien de l'admissibilité des trajectoires dans les limites des contraintes environnementales, comme le fait de rester dans les zones praticables. Elle prend la représentation de la carte m en entrée et génère un autre ensemble de codes latents partiels $z_m^{(n)}$.

Ces codes latents partiels issus des deux branches sont ensuite combinés par un produit élément par élément pour produire les codes latents finaux $z^{(n)}$ pour chaque trajectoire prédite. Cette combinaison permet que les trajectoires générées sont non seulement diversifiées, mais respectent également les contraintes physiques imposées par l'environnement de conduite, telles que les limites de la route et les marquages au sol.

Cette architecture permettant d'obtenir un ensemble de codes latents qui pourront ensuite être décodés en un ensemble diversifié de trajectoires futures doit être entraîné, et la fonction de coût créée pour cette occasion inclut deux composants qui reflètent les deux aspects que nous souhaitons promouvoir : la diversité et l'admissibilité.

1. L_{dpp} : Cette composante de la fonction de perte est basée sur les Processus Ponctuels Déterminantaux (DPPs), qui sont utilisés pour modéliser les corrélations négatives entre les éléments d'un ensemble. Dans ce contexte, le noyau DPP est adapté pour s'assurer que les points de terminaison des trajectoires prédites sont dispersés, en particulier dans la direction latérale, ce qui est crucial pour capturer différents comportements de direction.

2. L_{layout} : Ce terme pénalise les trajectoires qui violent les contraintes topologiques de la zone de conduite, comme celles qui dépassent les zones praticables. En incorporant cette perte, la DSF s'assure que les trajectoires générées sont non seulement diversifiées, mais aussi réalistes et applicables au scénario de conduite donné.

La fonction de coût globale pour l'entraînement de la DSF est une somme pondérée de ces deux composantes :

$$L_{dsf} = \lambda L_{dpp} + (1 - \lambda) L_{layout}$$

où λ est un paramètre qui contrôle le compromis entre la diversité et la qualité.

Expériences. Les résultats expérimentaux, basés sur le jeu de données nuScenes, montrent que l'architecture proposée permet non seulement d'améliorer la diversité des trajectoires générées, mais aussi de maintenir une bonne précision par rapport à la trajectoire réelle observée. Les expériences qualitatives mettent en évidence des améliorations significatives dans divers scénarios, y compris les intersections et les lignes droites, où le modèle DIVA génère des directions diverses et plausibles, contrairement aux modèles cVAE de base qui souffrent d'un effondrement des modes.

Pour évaluer l'efficacité des modèles proposés, la thèse utilise plusieurs jeux de données, dont le populaire nuScenes, et applique une série de métriques pour mesurer la diversité et la qualité des trajectoires générées. Les métriques incluent des mesures de la distance moyenne au sol (average displacement error, ADE) et la distance moyenne finale (final displacement error, FDE), qui évaluent la précision des trajectoires par rapport aux trajectoires réelles.

En outre, des métriques spécifiques à la diversité sont également employées, telles que la dispersion (spread) des trajectoires et la couverture des modes (mode coverage), qui mesurent à quel point les trajectoires générées couvrent l'éventail des possibilités représentées dans les données.

En conclusion, ce premier travail met en avant l'importance de la diversité dans les modèles génératifs pour la prédiction de trajectoires en conduite autonome. En combinant un modèle génératif sous-jacent avec un composant de diversité structuré, cette approche permet de surmonter les limitations des méthodes traditionnelles qui se concentrent sur les modes majoritaires, et de mieux capturer les

scénarios rares mais critiques. Les contributions de ce premier travail offrent une nouvelle perspective sur la manière d'intégrer efficacement la diversité dans les systèmes de conduite autonome, ouvrant ainsi la voie à des véhicules plus sûrs et plus fiables.

Découverte de nouveaux modes

Pour aller plus loin que la diversité représentée dans les données d'entraînement, nous avons élargi la problématique en partant du constat que dans tous les datasets, des trajectoires possibles mais non représentées pouvaient exister, et qu'elles seraient intéressantes à inclure dans la génération de trajectoires possibles. La découverte dans les modèles génératifs, lorsqu'il s'agit de générer des trajectoires admissibles mais absentes de la distribution d'entraînement, est une problématique d'extrapolation complexe. Contrairement à la simple diversité, où l'objectif est de couvrir un large éventail de scénarios probables en fonction des données existantes, la découverte vise à générer des trajectoires qui n'ont jamais été observées pendant l'entraînement, mais qui sont néanmoins plausibles et conformes aux contraintes de l'environnement routier. Ce concept est particulièrement pertinent dans le contexte de la conduite autonome, où des scénarios rares mais critiques, non représentés dans les données d'entraînement, peuvent avoir un impact significatif sur la sécurité.

La motivation principale derrière cette recherche est de combler les lacunes des modèles génératifs traditionnels qui, même lorsqu'ils sont optimisés pour la diversité, ne peuvent générer que des échantillons proches des trajectoires déjà observées. En d'autres termes, ces modèles sont limités par la distribution de données d'entraînement, ce qui signifie que des trajectoires importantes, mais non présentes dans ces données, ne sont jamais générées. Dans un contexte de conduite autonome, cela peut entraîner une préparation insuffisante face à des événements rares mais dangereux. La découverte permettrait d'anticiper de tels scénarios en créant des trajectoires admissibles mais jamais observées.

Dataset expérimental Afin de tester et de valider la faisabilité de la découverte, il est nécessaire de proposer la création d'un jeu de données synthétiques où certaines modalités de trajectoires (par exemple, les virages à gauche) sont délibérément omises pendant l'entraînement. L'objectif est alors de voir si le modèle peut générer ces trajectoires manquantes sans les avoir observées pendant l'entraînement. Ce défi est abordé à travers plusieurs étapes expérimentales, notamment la création d'un modèle de découverte qui utilise un schéma d'auto-étiquetage et des contraintes externes d'admissibilité pour encourager la génération de nouvelles trajectoires.

Approche proposée L'approche développée pour résoudre ce problème repose sur plusieurs innovations techniques. Premièrement, un modèle en une seule étape est proposé, où l'ensemble du modèle (encodeur,

générateur de codes latents et décodeur) est entraîné de manière conjointe. Cette architecture de bout en bout est essentielle pour permettre au décodeur de produire des échantillons qui s'éloignent de la distribution de formation traditionnelle. Contrairement à DIVA, où le générateur de codes latents peut être formé sur un backbone pré-entraîné, la découverte nécessite que le décodeur soit formé simultanément avec le générateur de codes latents pour apprendre à décoder les codes représentant de nouvelles trajectoires.

Deuxièmement, le lien direct entre les sorties de l'encodeur et les entrées du décodeur est coupé afin de limiter la dépendance excessive aux trajectoires passées, ce qui permet au modèle de se concentrer davantage sur la génération de nouvelles trajectoires plutôt que de simplement prédire des variations mineures de celles déjà observées. Cette étape est cruciale pour encourager l'exploration en dehors de la distribution d'entraînement.

Résultats Les résultats montrent que le modèle proposé est capable de générer des trajectoires non observées pendant l'entraînement, en particulier dans des configurations de routes complexes comme les intersections. Les expérimentations montrent que les gains en termes de couverture des scénarios critiques sont significatifs. Par exemple, le modèle est capable de générer des virages à gauche dans des situations où toutes les trajectoires dans le dataset d'entraînement allaient tout droit ou tournaient à droite. Cela démontre la capacité du modèle à extrapoler au-delà de la distribution d'entraînement tout en respectant les contraintes d'admissibilité.

Une analyse qualitative met en évidence des exemples concrets de trajectoires découvertes, y compris des manœuvres rares comme des demi-tours, qui bien que peu fréquentes, sont critiques pour la sécurité routière. Cette capacité à découvrir et à générer de telles trajectoires offre des perspectives intéressantes pour améliorer la robustesse des systèmes de conduite autonome face à des scénarios inattendus.

En conclusion, la découverte, en tant que forme avancée de diversité, est un outil puissant pour améliorer la sécurité des systèmes de conduite autonome. En permettant de générer des trajectoires admissibles mais non observées pendant l'entraînement, la découverte offre une nouvelle dimension à l'anticipation des événements rares et critiques. Bien que ce travail soit encore exploratoire, il jette les bases d'une nouvelle approche pour la prévision des trajectoires, allant au-delà des méthodes traditionnelles qui se limitent à la diversité au sein de la distribution d'entraînement. Les défis futurs incluront l'amélioration des mécanismes de sélection des trajectoires découvertes et l'intégration de ces méthodes dans des systèmes de conduite autonomes à grande échelle.

Conclusion et perspectives

Récapitulatif des contributions. L'un des apports majeurs de cette thèse est l'accent mis sur la diversité prédictive dans la génération de trajectoires. En contexte réel, les jeux de données présentent souvent des déséquilibres, avec une sur-représentation de certaines trajectoires, comme les trajets en ligne droite. Les modèles traditionnels ont tendance à se concentrer sur la prévision de ces trajectoires dominantes, laissant de côté des trajectoires moins probables mais tout aussi importantes pour la sécurité routière. En réponse à ce problème, cette thèse a proposé des mécanismes innovants, tels que l'intégration des processus ponctuels déterminantaux (DPPs), pour encourager la génération d'un ensemble diversifié de trajectoires futures.

Par ailleurs, l'approche développée dans le cadre de cette thèse ne se limite pas à la diversité des trajectoires, mais s'étend également à la découverte de nouvelles trajectoires, non observées dans les données d'entraînement. Cette capacité à générer des trajectoires "hors distribution" est essentielle pour traiter des scénarios rares mais critiques, qui ne sont pas capturés dans les jeux de données disponibles. Le travail sur la découverte a permis de démontrer qu'il est possible d'étendre la distribution générative au-delà de celle de l'entraînement, en utilisant des fonctions d'admissibilité externes pour guider la génération de trajectoires conformes aux contraintes de sécurité mais absentes des données d'entraînement.

Perspectives pour la découverte autonome. Un des défis soulevés par ce travail concerne la transition vers une découverte véritablement autonome, où le modèle serait capable de sélectionner et d'intégrer de nouvelles trajectoires sans supervision humaine directe. Actuellement, l'approche nécessite encore l'utilisation de la vérité terrain pour sélectionner les trajectoires à réintégrer dans le processus d'entraînement, ce qui limite l'autonomie du modèle. Ce travail propose plusieurs pistes pour surmonter cette limitation, notamment par le développement de fonctions de sélection basées sur des heuristiques spatiales et des poids d'attention croisée, qui pourraient permettre au modèle de reconnaître et de prioriser les trajectoires les plus novatrices sans dépendre des données d'entraînement.

L'approche décrite repose sur une architecture de type encodeur-décodeur, qui permet de créer un espace latent contrôlable où la diversité des trajectoires peut être modulée. Le décodeur est conçu de manière à n'utiliser que l'information contenue dans l'espace latent, garantissant ainsi que les nouvelles trajectoires générées s'éloignent progressivement de la distribution d'entraînement pour explorer des modalités absentes, mais néanmoins admissibles.

Défis et recommandations pour les travaux futurs. Cette thèse explore plusieurs façons de promouvoir la diversité dans les sorties des modèles génératifs, et ouvre plusieurs défis pour les recherches futures,

notamment l'amélioration des mécanismes de sélection autonomes et l'extension des modèles développés à d'autres domaines au-delà de la prédiction de trajectoires. Par exemple, l'intégration de ces techniques dans des systèmes de génération d'images ou d'autres types de données pourrait offrir des résultats intéressants, en particulier dans des contextes où la diversité et la découverte sont essentielles pour la robustesse du système. A titre d'ouverture, nous remarquons que les approches proposées dans le contexte de la prédiction de trajectoire peuvent en fait être adaptés à n'importe quel modèle génératif qui comporte un espace latent.

Bibliography

[Alahi et al., 2016] Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., and Savarese, S. (2016). Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Bai et al., 2022] Bai, X., Hu, Z., Zhu, X., Huang, Q., Chen, Y., Fu, H., and Tai, C.-L. (2022). Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1090–1099.

[Bansal et al., 2018] Bansal, M., Krizhevsky, A., and Ogale, A. (2018). Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *Robotics: Science and Systems (RSS)*.

[Bauchwitz and Cummings, 2020] Bauchwitz, B. and Cummings, M. (2020). Evaluating the reliability of tesla model 3 driver assist functions. *Duke University*, 1.

[Ben-Younes et al., 2022] Ben-Younes, H., Zablocki, E., Chen, M., Pérez, P., and Cord, M. (2022). Raising context awareness in motion forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4409–4418.

[Benfold and Reid, 2011] Benfold, B. and Reid, I. (2011). Stable multi-target tracking in real-time surveillance video. In *CVPR 2011*, pages 3457–3464. IEEE.

[Bock et al., 2020] Bock, J., Krajewski, R., Moers, T., Runde, S., Vater, L., and Eckstein, L. (2020). 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.

[Bojarski et al., 2016] Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., et al. (2016). End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.

[Borgefors, 1984] Borgefors, G. (1984). Distance transformations in arbitrary dimensions. *Graphical Models graphical Models and Image Processing computer Vision, Graphics, and Image Processing*, 27:321–345.

[Borodin and Rains, 2005] Borodin, A. and Rains, E. (2005). Eynard-mehta theorem, schur process, and their pfaffian analogs. *Journal of Statistical Physics*, 121:291–317.

[Breuer et al., 2020] Breuer, A., Termöhlen, J.-A., Homoceanu, S., and Fingscheidt, T. (2020). opendd: A large-scale roundabout drone dataset. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE.

[Bronstein et al., 2017] Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

[Caesar et al., 2020] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Calem et al., 2022] Calem, L., Ben-Younes, H., Pérez, P., and Thome, N. (2022). Diverse probabilistic trajectory forecasting with admissibility constraints. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 3478–3484. IEEE.

[Casas et al., 2020] Casas, S., Gulino, C., Suo, S., and Urtasun, R. (2020). The importance of prior knowledge in precise multimodal prediction. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2295–2302. IEEE.

[Casas et al., 2021] Casas, S., Sadat, A., and Urtasun, R. (2021). Mp3: A unified model to map, perceive, predict and plan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14403–14412.

[Celis et al., 2016] Celis, L. E., Deshpande, A., Kathuria, T., Straszak, D., and Vishnoi, N. K. (2016). On the complexity of constrained determinantal point processes. *arXiv preprint arXiv:1608.00554*.

[Chadebec and Allassonnière, 2022] Chadebec, C. and Allassonnière, S. (2022). A geometric perspective on variational autoencoders. *Advances in Neural Information Processing Systems*, 35:19618–19630.

[Chai et al., 2019] Chai, Y., Sapp, B., Bansal, M., and Anguelov, D. (2019). Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In *Conference on Robot Learning (CoRL)*.

[Chang et al., 2019] Chang, M.-F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., et al. (2019). Argoverse: 3d tracking and forecasting with rich

maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Chitta et al., 2021] Chitta, K., Prakash, A., and Geiger, A. (2021). Neat: Neural attention fields for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15793–15803.

[Cho et al., 2014] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[Corbière, 2022] Corbière, C. (2022). Apprentissage profond robuste pour la conduite autonome.

[Cui et al., 2021] Cui, A., Casas, S., Sadat, A., Liao, R., and Urtasun, R. (2021). Lookout: Diverse multi-future prediction and planning for self-driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16107–16116.

[Cui et al., 2019] Cui, H., Radosavljevic, V., Chou, F., Lin, T., Nguyen, T., Huang, T., Schneider, J., and Djuric, N. (2019). Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, May 20-24, 2019*, pages 2090–2096. IEEE.

[Dauner et al., 2023] Dauner, D., Hallgarten, M., Geiger, A., and Chitta, K. (2023). Parting with misconceptions about learning-based vehicle motion planning. *arXiv preprint arXiv:2306.07962*.

[Deo and Trivedi, 2018] Deo, N. and Trivedi, M. M. (2018). Convolutional social pooling for vehicle trajectory prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1468–1476.

[Deo et al., 2022] Deo, N., Wolff, E., and Beijbom, O. (2022). Multimodal trajectory prediction conditioned on lane-graph traversals. In *Conference on Robot Learning*, pages 203–212. PMLR.

[Dosovitskiy et al., 2017] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16.

[Elfeki et al., 2019] Elfeki, M., Couprie, C., Riviere, M., and Elhoseiny, M. (2019). Gdpp: Learning diverse generations using determinantal point processes. In *International Conference on Machine Learning (ICML)*.

[Fan et al., 2018] Fan, H., Zhu, F., Liu, C., Zhang, L., Zhuang, L., Li, D., Zhu, W., Hu, J., Li, H., and Kong, Q. (2018). Baidu apollo em motion planner. *arXiv preprint arXiv:1807.08048*.

[Gao et al., 2020] Gao, J., Sun, C., Zhao, H., Shen, Y., Anguelov, D., Li, C., and Schmid, C. (2020). Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11525–11533.

[Geiger et al., 2013] Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237.

[Gilles et al., 2022] Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B., and Moutarde, F. (2022). Gohome: Graph-oriented heatmap output for future motion estimation. In *2022 international conference on robotics and automation (ICRA)*, pages 9107–9114. IEEE.

[Giuliari et al., 2021] Giuliari, F., Hasan, I., Cristani, M., and Galasso, F. (2021). Transformer networks for trajectory forecasting. In *2020 25th international conference on pattern recognition (ICPR)*, pages 10335–10342. IEEE.

[Gong et al., 2014] Gong, B., Chao, W.-L., Grauman, K., and Sha, F. (2014). Diverse sequential subset selection for supervised video summarization. *Advances in Neural Information Processing Systems (NeurIPS)*.

[Goodfellow et al., 2020] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, 63(11):139–144.

[Goodfellow et al., 2014] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[Greer et al., 2021] Greer, R., Deo, N., and Trivedi, M. (2021). Trajectory prediction in autonomous driving with a lane heading auxiliary loss. *IEEE Robotics and Automation Letters*, 6(3):4907–4914.

[Gu et al., 2023] Gu, J., Hu, C., Zhang, T., Chen, X., Wang, Y., Wang, Y., and Zhao, H. (2023). Vip3d: End-to-end visual trajectory prediction via 3d agent queries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5496–5506.

[Guen and Thome, 2020] Guen, V. L. and Thome, N. (2020). Probabilistic time series forecasting with structured shape and temporal diversity. In *Neural Information Processing Systems (NeurIPS)*.

[H. Caesar, 2021] H. Caesar, J. Kabzan, K. T. e. a. (2021). Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. In *CVPR ADP3 workshop*.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

[Higgins et al., 2017] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. (2017). beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations (ICLR)*.

[Ho et al., 2020] Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

[Hong and Nenkova, 2014] Hong, K. and Nenkova, A. (2014). Improving the estimation of word importance for news multi-document summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*.

[Hu et al., 2023] Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., Wang, W., et al. (2023). Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17853–17862.

[Huang et al., 2020a] Huang, X., McGill, S. G., DeCastro, J. A., Fletcher, L., Leonard, J. J., Williams, B. C., and Rosman, G. (2020a). Diversitygan: Diversity-aware vehicle motion prediction via latent semantic sampling. *IEEE Robotics Autom. Lett.*, 5(4):5089–5096.

[Huang et al., 2020b] Huang, X., McGill, S. G., DeCastro, J. A., Fletcher, L., Leonard, J. J., Williams, B. C., and Rosman, G. (2020b). Diversitygan: Diversity-aware vehicle motion prediction via latent semantic sampling. *IEEE Robotics and Automation Letters*, 5(4):5089–5096.

[Ioffe and Szegedy, 2015] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning (ICML)*.

[Ivanovic and Pavone, 2019] Ivanovic, B. and Pavone, M. (2019). The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2375–2384.

[Janai et al., 2020] Janai, J., Güney, F., Behl, A., Geiger, A., et al. (2020). Computer vision for autonomous vehicles: Problems, datasets and state of the art. *Foundations and Trends® in Computer Graphics and Vision*, 12(1–3):1–308.

[Kalman, 1960] Kalman, R. E. (1960). A new approach to linear filtering and prediction problems.

[Kendall et al., 2019] Kendall, A., Hawke, J., Janz, D., Mazur, P., Reda, D., Allen, J.-M., Lam, V.-D., Bewley, A., and Shah, A. (2019). Learning to drive in a day. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8248–8254. IEEE.

[Kim et al., 2021] Kim, B., Park, S. H., Lee, S., Khoshimjonov, E., Kum, D., Kim, J., Kim, J. S., and Choi, J. W. (2021). Lapred: Lane-aware prediction of multi-modal future trajectories of dynamic agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14636–14645.

[Kingma and Welling, 2014] Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*.

[Kosaraju et al., 2019] Kosaraju, V., Sadeghian, A., Martín-Martín, R., Reid, I., Rezatofighi, H., and Savarese, S. (2019). Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. *Advances in Neural Information Processing Systems*, 32.

[Krajewski et al., 2020] Krajewski, R., Moers, T., Bock, J., Vater, L., and Eckstein, L. (2020). The round dataset: A drone dataset of road user trajectories at roundabouts in germany. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE.

[Kulesza and Taskar, 2012] Kulesza, A. and Taskar, B. (2012). Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*.

[Lee et al., 2017] Lee, N., Choi, W., Vernaza, P., Choy, C. B., Torr, P. H., and Chandraker, M. (2017). Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Lerner et al., 2007] Lerner, A., Chrysanthou, Y., and Lischinski, D. (2007). Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library.

[Li et al., 2020] Li, Y., Xu, X., Xiao, J., Li, S., and Shen, H. T. (2020). Adaptive square attack: Fooling autonomous cars with adversarial traffic signs. *IEEE Internet of Things Journal*, 8(8):6337–6347.

[Li et al., 2022] Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., and Dai, J. (2022). Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *European conference on computer vision*, pages 1–18. Springer.

[Liang et al., 2020a] Liang, J., Jiang, L., Murphy, K., Yu, T., and Hauptmann, A. (2020a). 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.

[Liang et al., 2020b] Liang, M., Yang, B., Hu, R., Chen, Y., Liao, R., Feng, S., and Urtasun, R. (2020b). 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.

[Liu et al., 2020] Liu, R., Wang, J., and Zhang, B. (2020). High definition map for automated driving: Overview and analysis. *The Journal of Navigation*, 73(2):324–341.

[Liu et al., 2023] Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D. L., and Han, S. (2023). Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2774–2781. IEEE.

[Locatello et al., 2019] Locatello, F., Bauer, S., Lucic, M., Raetsch, G., Gelly, S., Schölkopf, B., and Bachem, O. (2019). Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pages 4114–4124. PMLR.

[Lucas et al., 2019] Lucas, J., Tucker, G., Grosse, R., and Norouzi, M. (2019). Understanding posterior collapse in generative latent variable models. In *International Conference on Learning Representations (ICLR)*.

[Ma et al., 2021] Ma, Y. J., Inala, J. P., Jayaraman, D., and Bastani, O. (2021). Likelihood-based diverse sampling for trajectory forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13279–13288.

[Macchi, 1975] Macchi, O. (1975). The coincidence approach to stochastic point processes. *Advances in Applied Probability*.

[Malinin et al., 2021] Malinin, A., Band, N., Chesnokov, G., Gal, Y., Gales, M. J., Noskov, A., Ploskonosov, A., Prokhorenkova, L., Provilkov, I., Raina, V., et al. (2021). Shifts: A dataset of real distributional shift across multiple large-scale tasks. *arXiv preprint arXiv:2107.07455*.

[Mangalam et al., 2020] Mangalam, K., Girase, H., Agarwal, S., Lee, K.-H., Adeli, E., Malik, J., and Gaidon, A. (2020). It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 759–776. Springer.

[Mao et al., 2021] Mao, J., Niu, M., Jiang, C., Liang, H., Chen, J., Liang, X., Li, Y., Ye, C., Zhang, W., Li, Z., et al. (2021). One million scenes for autonomous driving: Once dataset. *arXiv preprint arXiv:2106.11037*.

[Marchant and Lindor, 2012] Marchant, G. E. and Lindor, R. A. (2012). The coming collision between autonomous vehicles and the liability system. *Santa Clara L. Rev.*, 52:1321.

[Mariet et al., 2019] Mariet, Z., Ovadia, Y., and Snoek, J. (2019). Dppnet: Approximating determinantal point processes with deep networks. In *Neural Information Processing Systems (NeurIPS)*.

[Matthey et al., 2017] Matthey, L., Higgins, I., Hassabis, D., and Lerchner, A. (2017). dsprites: Disentanglement testing sprites dataset.

[McAllister et al., 2017] McAllister, R., Gal, Y., Kendall, A., Van Der Wilk, M., Shah, A., Cipolla, R., and Weller, A. (2017). Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning. International Joint Conferences on Artificial Intelligence, Inc.

[Montero et al., 2022] Montero, M. L., Bowers, J. S., Costa, R. P., Ludwig, C. J., and Malhotra, G. (2022). Lost in latent space: Disentangled models and the challenge of combinatorial generalisation. In *Neural Information Processing Systems (NeurIPS)*.

[Montero et al., 2020] Montero, M. L., Ludwig, C. J., Costa, R. P., Malhotra, G., and Bowers, J. (2020). The role of disentanglement in generalisation. In *International Conference on Learning Representations*.

[Narayanan et al., 2021] Narayanan, S., Moslemi, R., Pittaluga, F., Liu, B., and Chandraker, M. (2021). Divide-and-conquer for lane-aware diverse trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Naumann et al., 2023] Naumann, A., Hertlein, F., Grimm, D., Zipfl, M., Thoma, S., Rettinger, A., Halilaj, L., Luettin, J., Schmid, S., and Caesar, H. (2023). Lanelet2 for nuscenes: Enabling spatial semantic relationships and diverse map-based anchor paths. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3247–3256.

[Nishimura et al., 2023] Nishimura, H., Mercat, J., Wulfe, B., McAllister, R. T., and Gaidon, A. (2023). Rap: Risk-aware prediction for robust planning. In *Conference on Robot Learning*, pages 381–392. PMLR.

[Notin et al., 2021] Notin, P., Hernández-Lobato, J. M., and Gal, Y. (2021). Improving black-box optimization in vae latent space using decoder uncertainty. *Advances in Neural Information Processing Systems*, 34:802–814.

[Pang et al., 2023] Pang, Z., Ramanan, D., Li, M., and Wang, Y.-X. (2023). Streaming motion forecasting for autonomous driving. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7407–7414. IEEE.

[Park et al., 2020] Park, S. H., Lee, G., Seo, J., Bhat, M., Kang, M., Francis, J., Jadhav, A., Liang, P. P., and Morency, L.-P. (2020). Diverse and admissible trajectory forecasting through multimodal context understanding. In *European Conference on Computer Vision (ECCV)*.

[Pellegrini et al., 2009] Pellegrini, S., Ess, A., Schindler, K., and Van Gool, L. (2009). You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*, pages 261–268. IEEE.

[Phan-Minh et al., 2020] Phan-Minh, T., Grigore, E. C., Boulton, F. A., Beijbom, O., and Wolff, E. M. (2020). Covernet: Multimodal behavior prediction using trajectory sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Ramasinghe et al., 2021] Ramasinghe, S., Ranasinghe, K., Khan, S., Barnes, N., and Gould, S. (2021). Conditional generative modeling via learning the latent space. In *International Conference on Learning Representations (ICLR)*.

[Razavi et al., 2019] Razavi, A., van den Oord, A., Poole, B., and Vinyals, O. (2019). Preventing posterior collapse with delta-VAEs. In *International Conference on Learning Representations (ICLR)*.

[Rezende and Mohamed, 2015a] Rezende, D. and Mohamed, S. (2015a). Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR.

[Rezende and Mohamed, 2015b] Rezende, D. and Mohamed, S. (2015b). Variational inference with normalizing flows. In *International Conference on Machine Learning (ICML)*.

[Rhinehart et al., 2018] Rhinehart, N., Kitani, K., and Vernaza, P. (2018). R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting. In *European Conference on Computer Vision (ECCV)*.

[Rhinehart et al., 2019] Rhinehart, N., McAllister, R., Kitani, K., and Levine, S. (2019). Precog: Prediction conditioned on goals in visual multi-agent settings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2821–2830.

[Robicquet et al., 2016] Robicquet, A., Sadeghian, A., Alahi, A., and Savarese, S. (2016). Learning social etiquette: Human trajectory understanding in crowded scenes. In *European conference on computer vision (ECCV)*.

[Robinson et al., 2019] Robinson, J., Sra, S., and Jegelka, S. (2019). Flexible modeling of diversity with strongly log-concave distributions. In *Neural Information Processing Systems (NeurIPS)*.

[Roddenberry et al., 2021] Roddenberry, T. M., Glaze, N., and Segarra, S. (2021). Principled simplicial neural networks for trajectory prediction. In *International Conference on Machine Learning*, pages 9020–9029. PMLR.

[Rombach et al., 2022] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695.

[Sadeghian et al.,] Sadeghian, A., Kosaraju, V., Gupta, A., Savarese, S., and Alahi, A. Trajnet: Towards a benchmark for human trajectory prediction. arxiv 2018. *Google Scholar*.

[Sadeghian et al., 2019] Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Rezatofighi, H., and Savarese, S. (2019). Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1349–1358.

[Salzmann et al., 2020] Salzmann, T., Ivanovic, B., Chakravarty, P., and Pavone, M. (2020). Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control. In *European Conference on Computer Vision (ECCV)*.

[Schmerling et al., 2018] Schmerling, E., Leung, K., Vollprecht, W., and Pavone, M. (2018). Multimodal probabilistic model-based planning for human-robot interaction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3399–3406. IEEE.

[Shawe-Taylor et al., 2004] Shawe-Taylor, J., Cristianini, N., et al. (2004). *Kernel methods for pattern analysis*. Cambridge university press.

[Sun et al., 2020] Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al. (2020). Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454.

[Tang and Salakhutdinov, 2019] Tang, C. and Salakhutdinov, R. R. (2019). Multiple futures prediction. *Advances in neural information processing systems*, 32.

[Tas et al., 2018] Tas, Ö. S., Hauser, F., and Stiller, C. (2018). Decision-time postponing motion planning for combinatorial uncertain maneuvering. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2419–2425. IEEE.

[Varghese et al., 2015] Varghese, J. Z., Boone, R. G., et al. (2015). Overview of autonomous vehicle sensors and systems. In *International Conference on Operations Excellence and Service Engineering*, volume 2015. sn.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*.

[Wang et al., 2007] Wang, J. M., Fleet, D. J., and Hertzmann, A. (2007). Gaussian process dynamical models for human motion. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):283–298.

[Wang et al., 2018] Wang, S., Suo, S., Ma, W.-C., Pokrovsky, A., and Urtasun, R. (2018). Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2589–2597.

[Weng et al., 2020a] Weng, X., Yuan, Y., and Kitani, K. (2020a). End-to-end 3d multi-object tracking and trajectory forecasting. *arXiv preprint arXiv:2008.11598*.

[Weng et al., 2020b] Weng, X., Yuan, Y., and Kitani, K. (2020b). Joint 3d tracking and forecasting with graph neural network and diversity sampling. *arXiv preprint arXiv:2003.07847*, 2(6.2):1.

[Weng et al., 2021] Weng, X., Yuan, Y., and Kitani, K. (2021). Ptp: Parallelized tracking and prediction with graph neural networks and diversity sampling. In *International Conference on Robotics and Automation (ICRA)*.

[Wiederer et al., 2023] Wiederer, J., Schmidt, J., Kressel, U., Dietmayer, K., and Belagiannis, V. (2023). Joint out-of-distribution detection and uncertainty estimation for trajectory prediction. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5487–5494. IEEE.

[Williams, 1998] Williams, C. K. (1998). Prediction with gaussian processes: From linear regression to linear prediction and beyond. In *Learning in graphical models*, pages 599–621. Springer.

[Xu et al., 2023] Xu, Y., Chambon, L., Éloi Zablocki, Chen, M., Alahi, A., Cord, M., and Pérez, P. (2023). Towards motion forecasting with real-world perception inputs: Are end-to-end approaches competitive?

[Yuan and Kitani, 2020a] Yuan, Y. and Kitani, K. (2020a). Diverse trajectory forecasting with determinantal point processes. In *International Conference on Learning Representations (ICLR)*.

[Yuan and Kitani, 2020b] Yuan, Y. and Kitani, K. (2020b). Dlow: Diversifying latent flows for diverse human motion prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[Yuan et al., 2021] Yuan, Y., Weng, X., Ou, Y., and Kitani, K. M. (2021). Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9813–9823.

[Zeng et al., 2019a] Zeng, W., Luo, W., Suo, S., Sadat, A., Yang, B., Casas, S., and Urtasun, R. (2019a). End-to-end interpretable neural motion planner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8660–8669.

[Zeng et al., 2019b] Zeng, W., Luo, W., Suo, S., Sadat, A., Yang, B., Casas, S., and Urtasun, R. (2019b). End-to-end interpretable neural motion planner. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8660–8669. Computer Vision Foundation / IEEE.

[Zhan et al., 2016] Zhan, W., Liu, C., Chan, C.-Y., and Tomizuka, M. (2016). A non-conservatively defensive strategy for urban autonomous driving. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 459–464. IEEE.

[Zhang et al., 2013] Zhang, S., Deng, W., Zhao, Q., Sun, H., and Litkouhi, B. (2013). Dynamic trajectory planning for vehicle autonomous driving. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE.

[Zhao et al., 2019] Zhao, T., Xu, Y., Monfort, M., Choi, W., Baker, C., Zhao, Y., Wang, Y., and Wu, Y. N. (2019). Multi-agent tensor fusion for contextual trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Zhou et al., 2012] Zhou, B., Wang, X., and Tang, X. (2012). Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2871–2878. IEEE.

Insert here an eventual dedication ...

Remerciements

Remerciements Nicolas et Patrick, Valeo.ai et CNAM

Au delà de toutes ces choses dont je vous remercie et que vous savez déjà, j'aimerais remercier chacun d'entre vous pour quelque chose en particulier, parmi tout ce que vous m'avez apporté:

Patrick Pérez et Nicolas Thome, *qui ont dit oui*,

Yann Auxéméry, *pour le saumon*,

Florent Bartoccioni, *pour avoir partagé la team forecasting avec moi*,

Hedi Ben-Younes, *pour le début de ma thèse*,

Samia Bouzefrane, *pour son oreille attentive*,

Andrei Bursuc, *pour "je suis content que tu sois revenue"*,

Amaia Cardiel, *pour son style fou et ses lapins*,

Mickaël Chen, *qui m'a guidée avec bienveillance sur mes directions de recherche parfois chaotiques*,

Charles Corbière, *pour son expérience*,

Marie Doloir, *pour avoir résolu avec une grande patience tous mes problèmes administratifs*,

Perla Doubinsky, *pour nos discussion sur les GANs*,

Axelle Elrikh, *pour m'avoir dit ce que j'avais besoin d'entendre*,

Pegah Khayatan , *pour avoir discuté planner et ses roses de Noël*,

Marc Lafon, *pour nos discussions sur les modèles à énergie*,

Renaud Marlet, *pour m'avoir fait découvrir les meilleurs Pannetone du monde*,

Tetiana Martinyuk, *pour ses conseils en vyshyvanka*,

Björn Michele, *pour avoir trinqué avec nous*,

Olivier Petit, *pour en être revenu*,

Elias Ramzi, *pour ses bonnes questions*,

Claire Ryckmans, *pour avoir cherché des solutions*,

Corentin Sautier, *pour avoir marché avec moi*,

Oriane Siméoni, *pour m'avoir patiemment écoutée autour d'un wheel cake*,

Sophia Sirko - Galouchenko, *pour les soirées jeux*,

Maïté Sylla, *pour sa bienveillance*,

Loïc Themyr, *pour avoir porté la chaise de bureau dans tout le métro*,

Tout Valeo.ai, *pour leurs connaissances, leurs questions, leur écoute*,

Toute l'équipe du CNAM, *pour nos déménagements intempestifs et les raclettes chez Perla*,

Léa et Lucie Calem, *pour nos conversations nocturnes*,

Sophie et Lionel Calem, *pour leur soutien sans faille*,

Jean-Baptiste Lorteau, *pour avoir tenu presque jusqu'au bout*

Appendix A

Additional DIVA visualizations

To illustrate the benefit of our compound angular/distance kernel K introduced in Chapter 3, we compare its behaviour in Figure A.1, to the ones of distance-only and angle-only kernels in various situations. We see that the distance kernel produces temporally "dilated" trajectories (speed variations along a single trajectory) as this is the easiest way to maximize the diversity without exiting the drivable area; The angular kernel exhibits a "two-mode collapse" regime as angles are computed pairwise; In contrast, our hybrid kernel samples well both speeds and directions while respecting environment constraints.

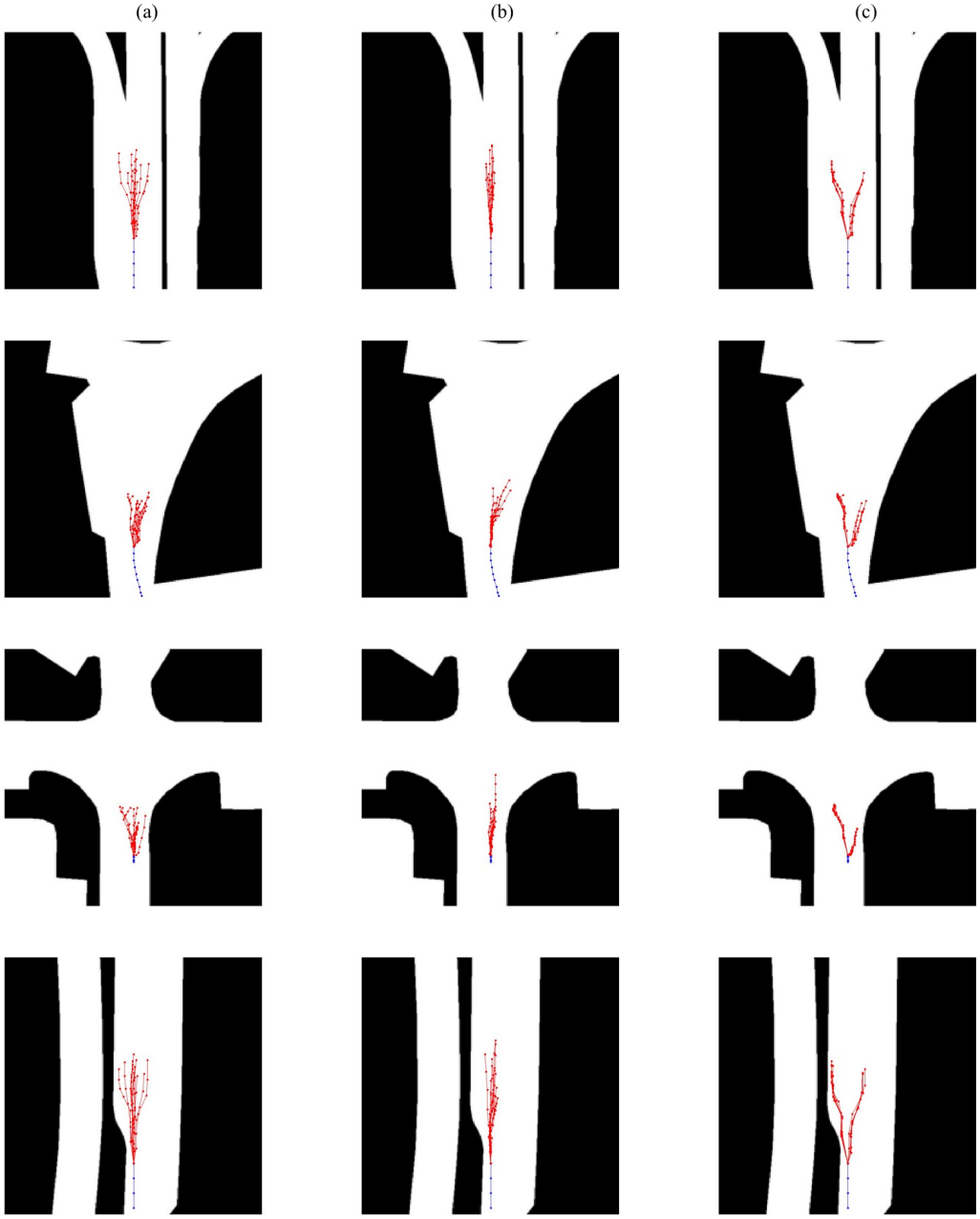


Figure A.1: **Effects of different kernels in various layouts**. (a) Our compound angular/distance kernel strikes a good balance between speed and direction sampling; (b) A distance L2 kernel allows speed sampling only; (c) An angular kernel samples two modes of driving direction.

Appendix B

Generating out of distribution trajectories

B.1 Effect of β on the diversity of a cVAE model

The baseline cVAE model outlined in section 4.3.1 doesn't exhibit much diversity let alone discovery. We studied the effect of the weight on the regularization term β in equation 4.1 to examine whether it had an effect on diversity, and found it didn't. Below are generated trajectories from various cVAE models trained with different β values.

B.2 Extrapolation for 2-step model

Below are the additional figures showing the results of generating trajectories for unknown latent codes z^u outside of the training range accepted by the decoder. Overall, the decoder produces trajectories that still look like trajectories, but essentially the same as the closest known latent code z^k. No significant further extrapolation is made even when the latent codes are more than twice the value of the original range. The only notable exception is for the straight trajectories (B.8 and B.9 especially) where it goes slightly left, but still fails to significantly move the endpoint of the generated trajectory.

In the following figures B.2 to B.9, z denotes the latent codes, subscript the dimension (as the example latent codes are 2-dimensional for visualization purposes), superscript u for unknown codes and k for known ones.

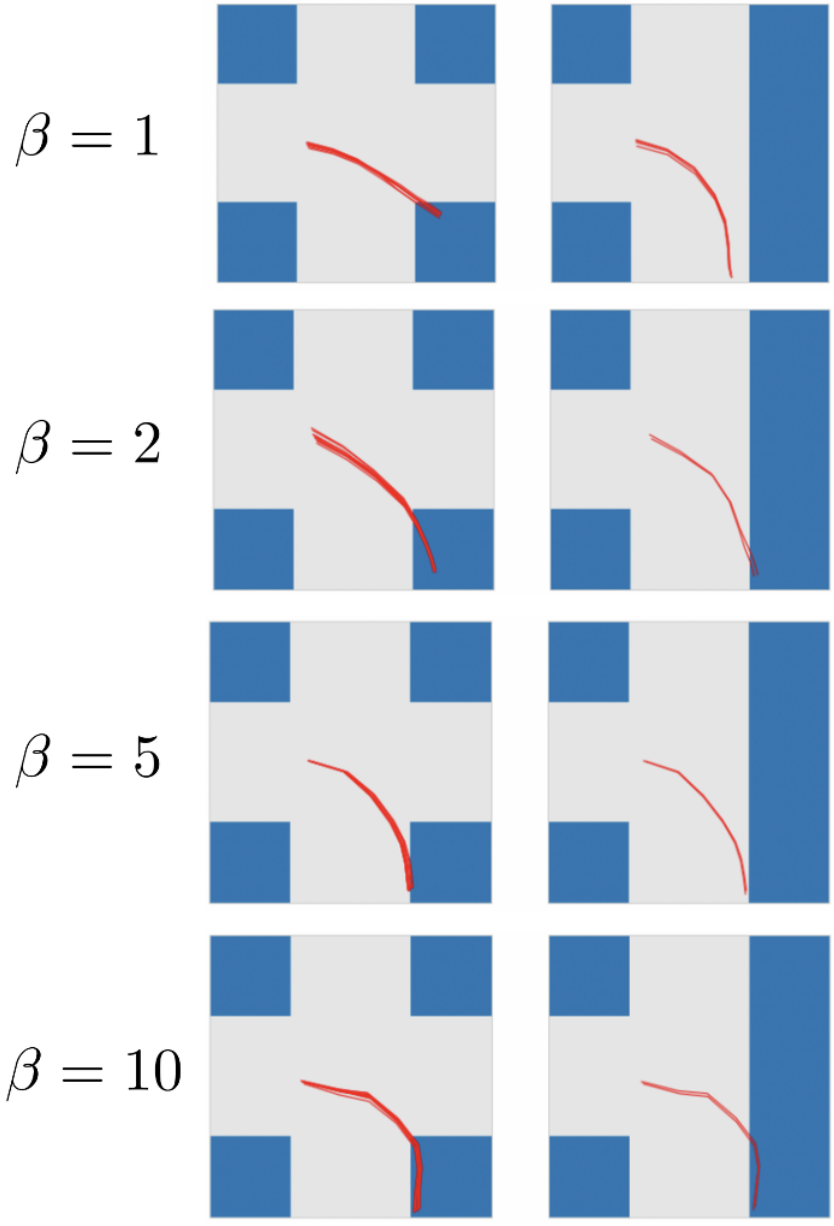


Figure B.1: Effect of β for cVAE diversity

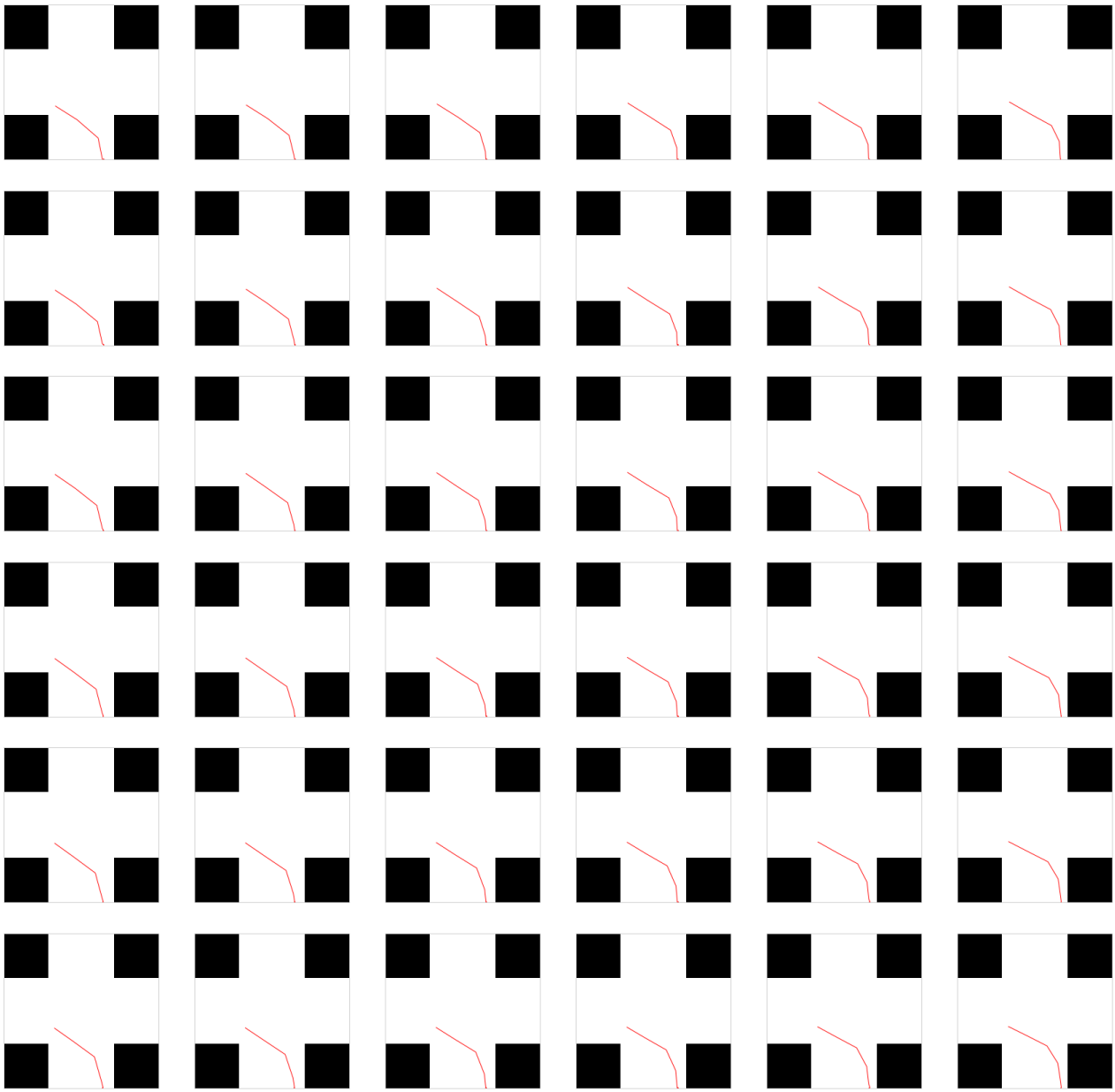


Figure B.2: Extrapolation for cross layout for $z_1^u < z_1^k$ and $z_2^u < z_2^k$.

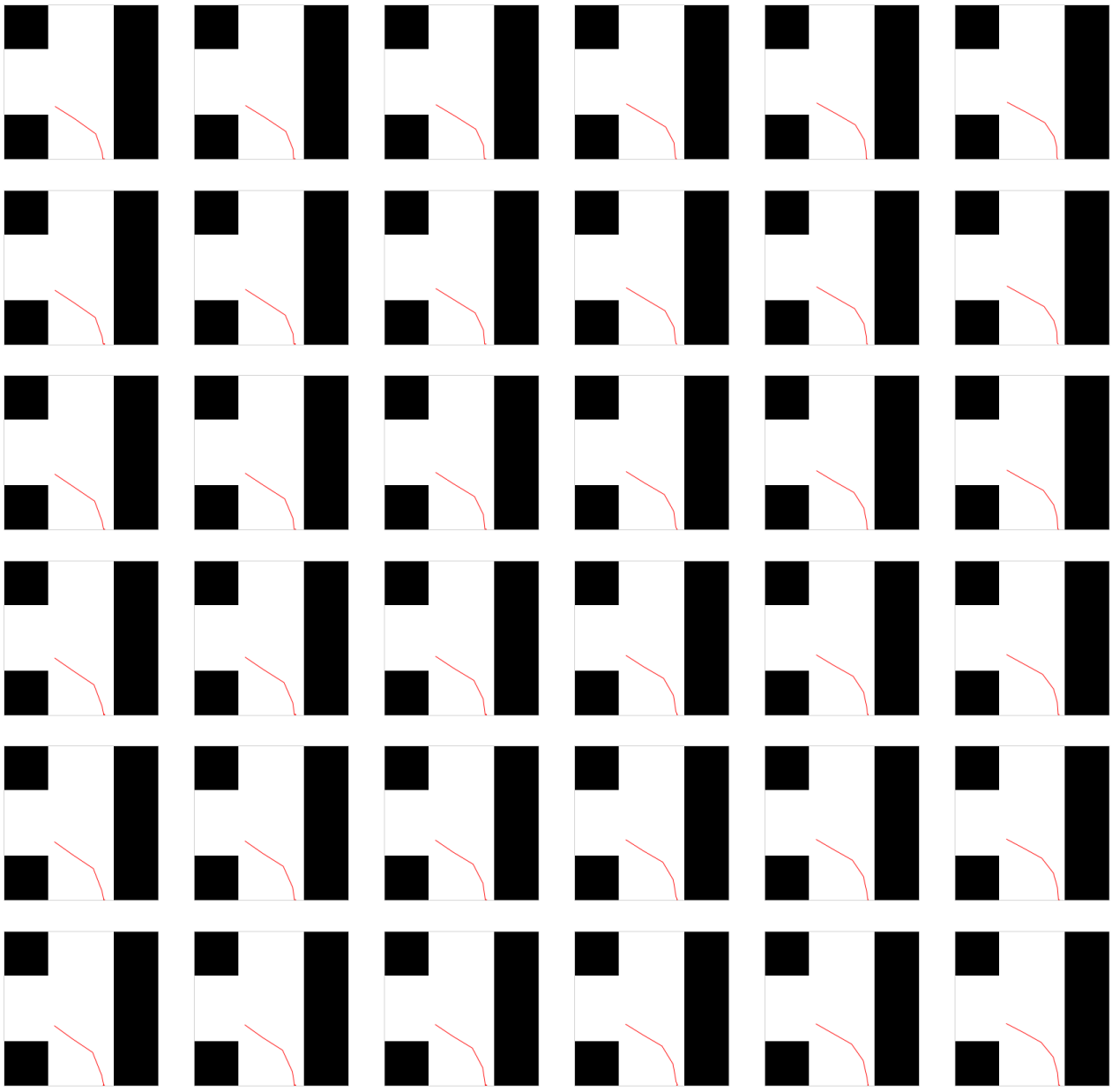


Figure B.3: Extrapolation for t-shaped layout for $z_1^u < z_1^k$ and $z_2^u < z_2^k$.

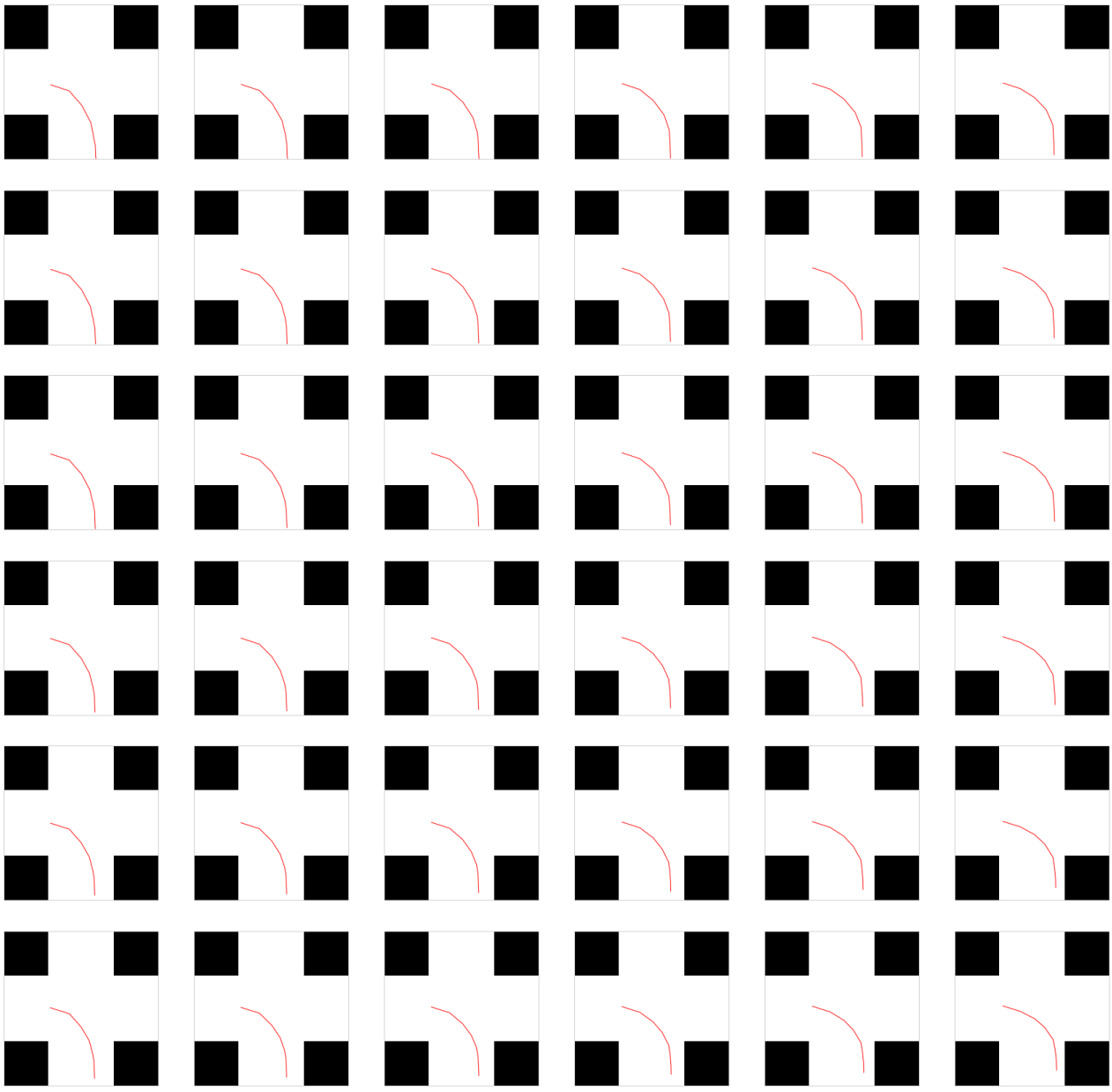


Figure B.4: Extrapolation for cross layout for $z_1^u > z_1^k$ and $z_2^u < z_2^k$.

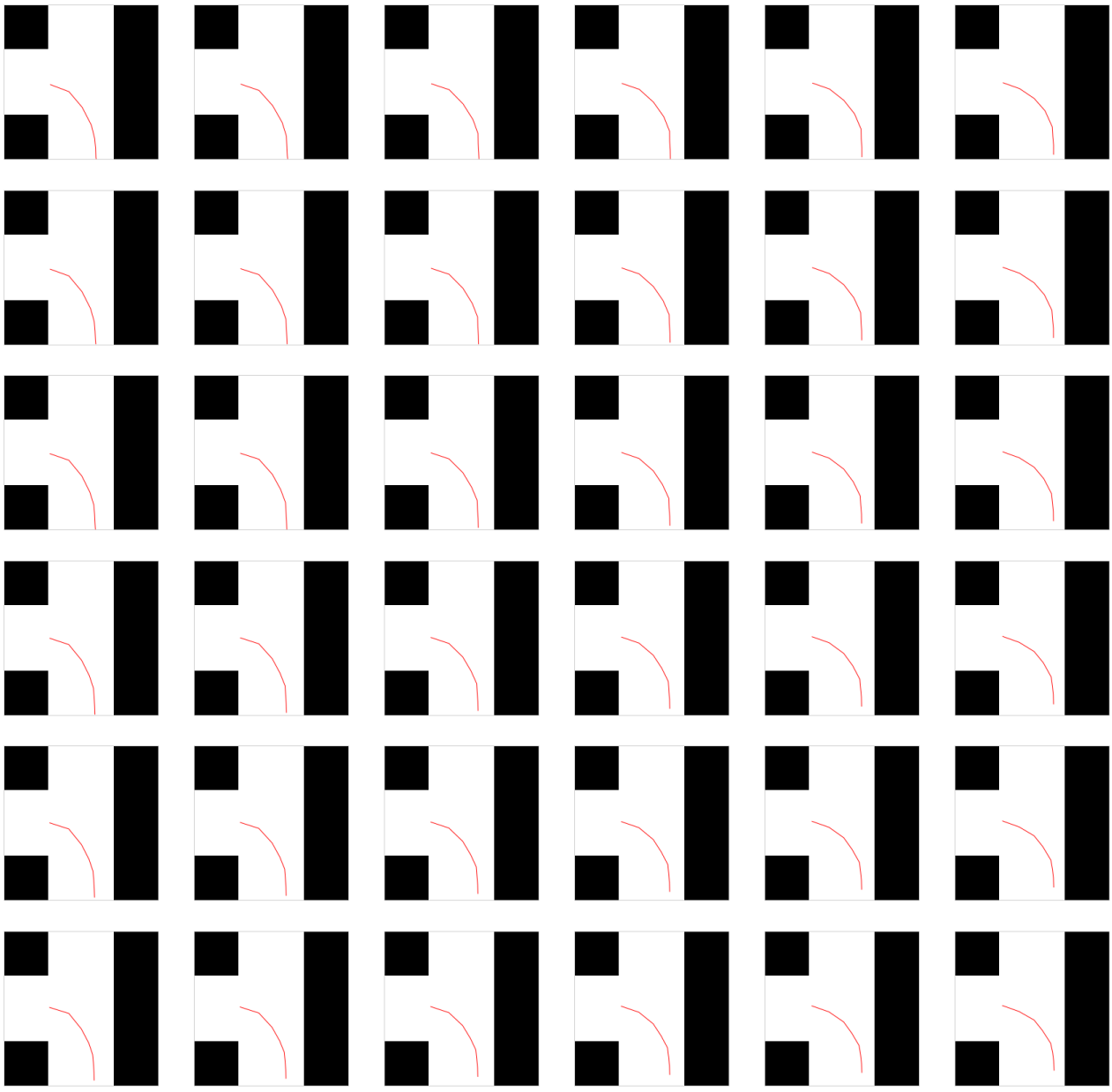


Figure B.5: Extrapolation for t-shaped layout for $z_1^u > z_1^k$ and $z_2^u < z_2^k$.

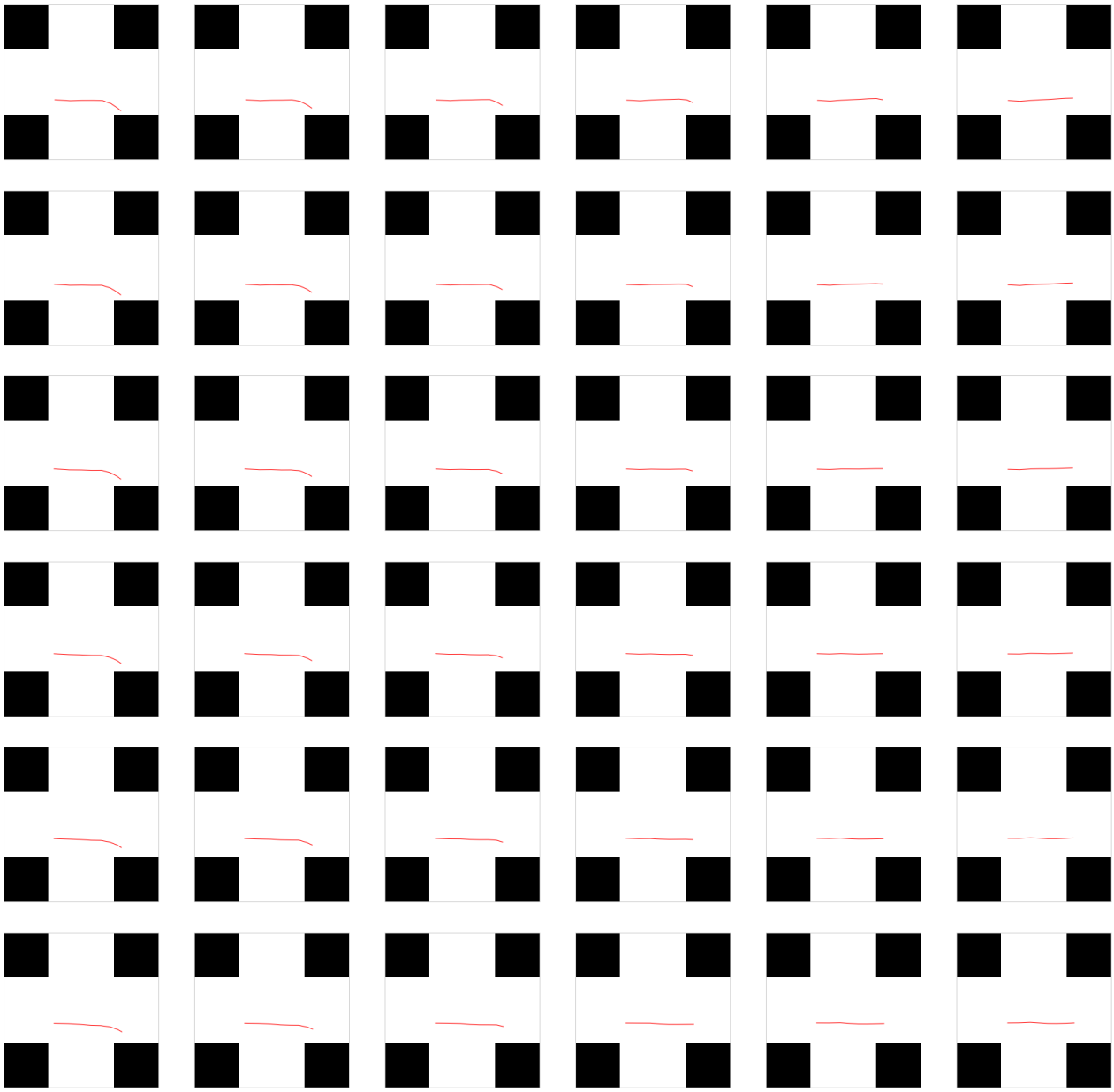


Figure B.6: Extrapolation for cross layout for $z_1^u < z_1^k$ and $z_2^u > z_2^k$.

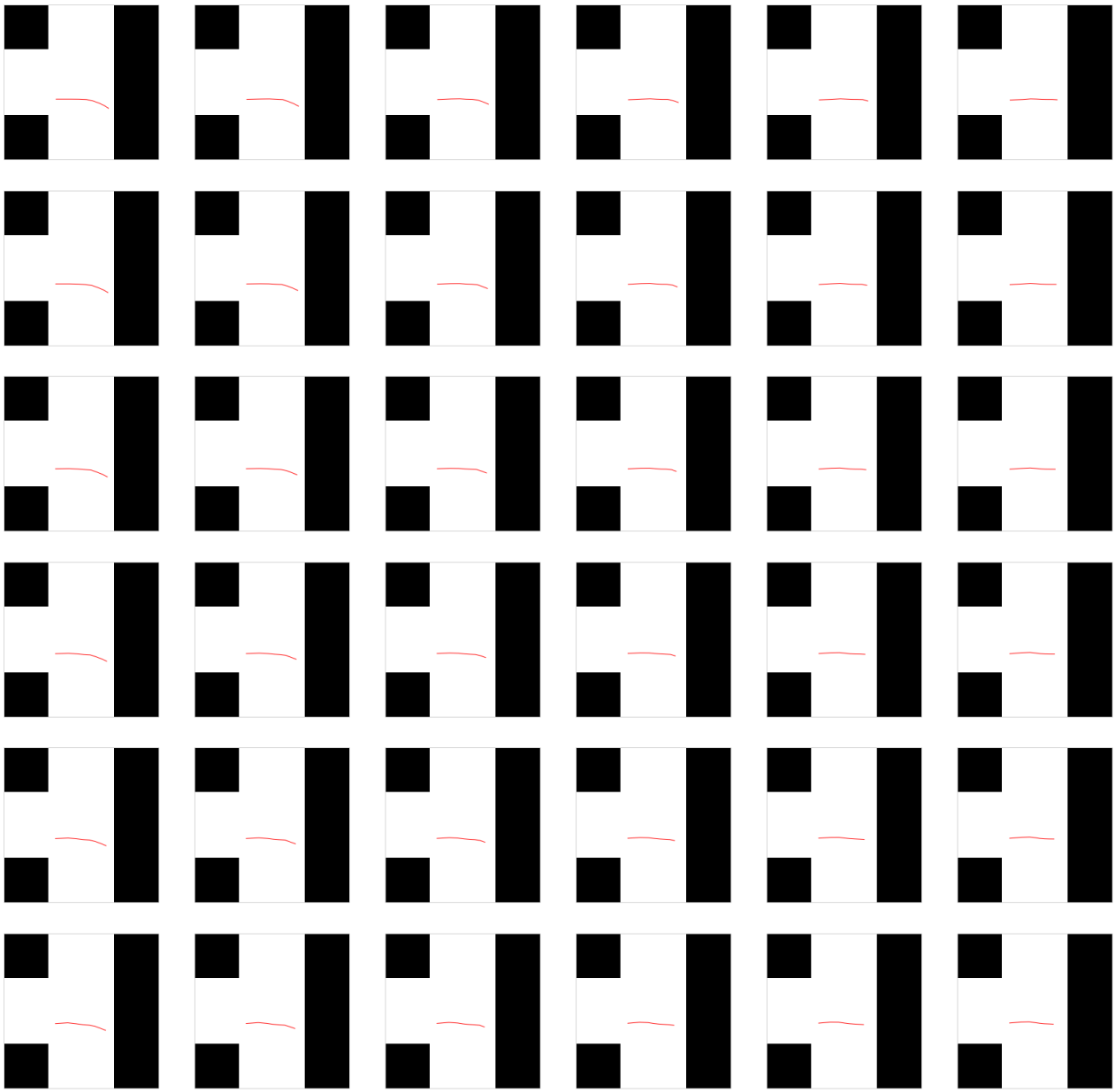


Figure B.7: Extrapolation for t-shaped layout for $z_1^u < z_1^k$ and $z_2^u > z_2^k$.

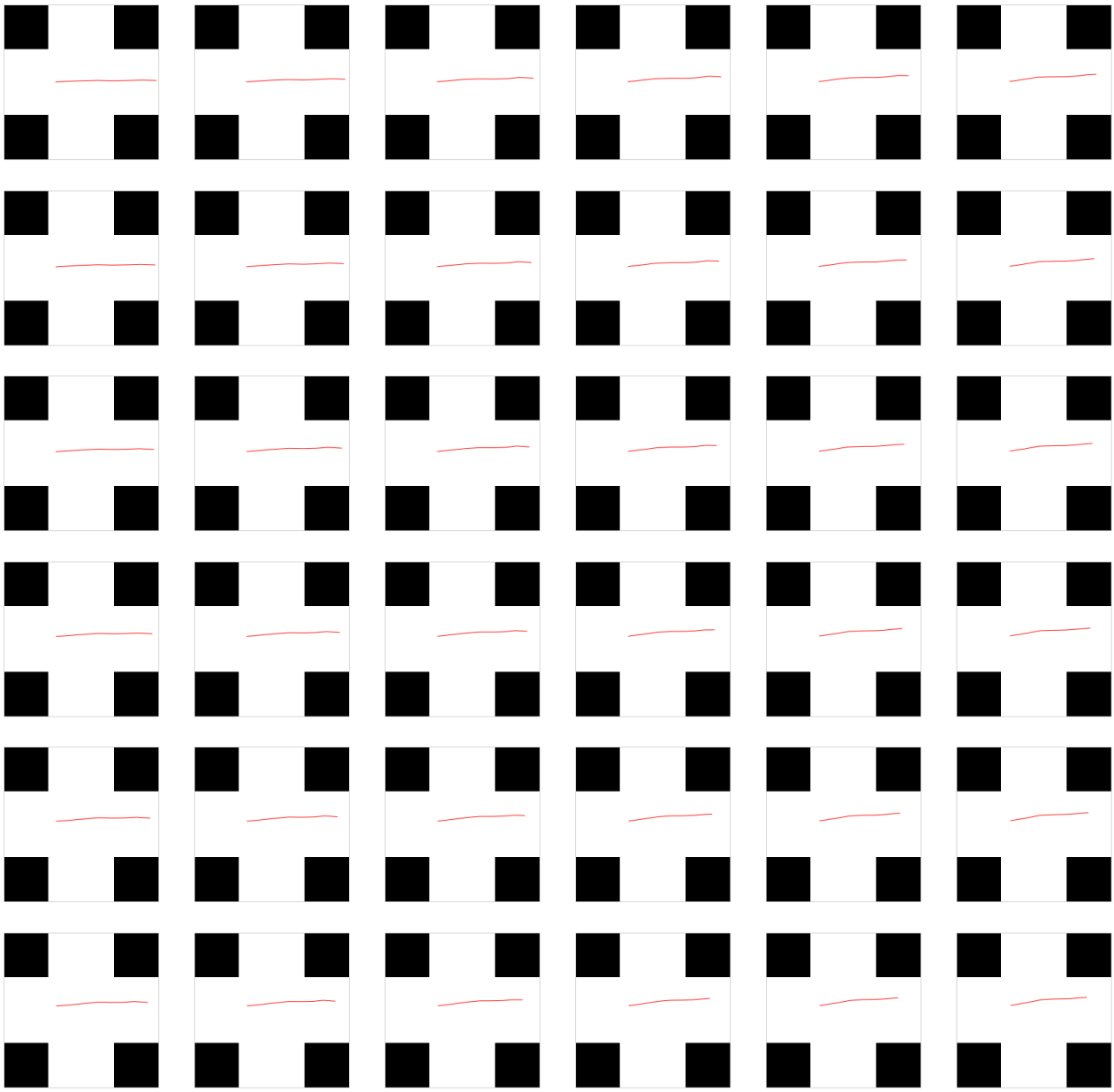


Figure B.8: Extrapolation for cross layout for $z_1^u > z_1^k$ and $z_2^u > z_2^k$.

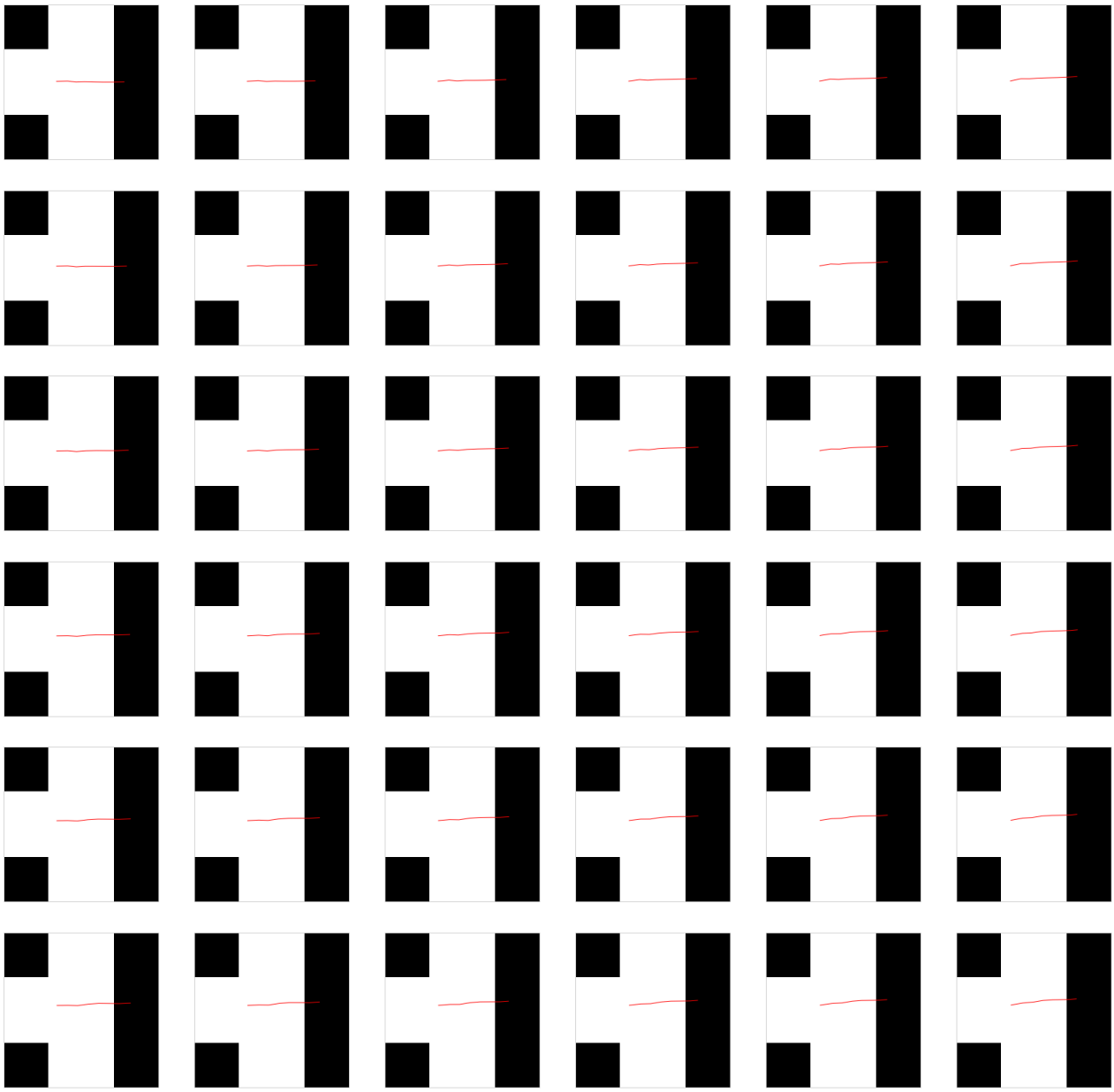


Figure B.9: Extrapolation for t-shaped layout for $z_1^u > z_1^k$ and $z_2^u > z_2^k$.