Journal of Artificial Intelligence Research 77 (2023) 1697-1772 Submitted 03/2023; published 08/2023

Prediction of Social Dynamic Agents and Long-Tailed Learning Challenges: A Survey

Divya Thuremella DIVYA@ROBOTS.OX.AC.UK
Lars Kunze LARS@ROBOTS.OX.AC.UK
Department of Engineering Science, University of Oxford, Parks Rd
Oxford OX13PJ, UK

Abstract

Autonomous robots that can perform common tasks like driving, surveillance, and chores have the biggest potential for impact due to frequency of usage, and the biggest potential for risk due to direct interaction with humans. These tasks take place in open-ended environments where humans socially interact and pursue their goals in complex and diverse ways. To operate in such environments, such systems must predict this behaviour, especially when the behavior is unexpected and potentially dangerous. Therefore, we summarize trends in various types of tasks, modeling methods, datasets, and social interaction modules aimed at predicting the future location of dynamic, socially interactive agents. Furthermore, we describe long-tailed learning techniques from classification and regression problems that can be applied to prediction problems. To our knowledge this is the first work that reviews social interaction modeling within prediction, and long-tailed learning techniques within regression and prediction.

1. Introduction

Autonomous robots are gaining the ability to reliably perceive the static objects and dynamic agents around them (Balasubramaniam & Pasricha, 2022), but in order to safely plan a future path without colliding with these obstacles, they need to be able to predict the future location of the dynamic agents. This is especially difficult in open-ended environments like roads and homes where dynamic agents can move around and interact with each other in an infinite variety of ways.

However, especially in open-ended environments with much possibility, a majority of agents execute the same simple actions like standing still or walking in a straight line, while few execute various complicated actions, like opening a car door or stopping to talk to a friend. This results in the situation depicted in Figure 1, where the model learns the majority common examples more easily than the few rare cases, resulting in a feature embedding space where uncommon paths get lost within the many embeddings of well-frequented paths and are therefore less easily identified and predicted accurately.

Furthermore, dynamic agents must be considered not just as individuals, but as parts of a social system in which future actions are contingent on others' intentions. Studying the many ways these intentions are communicated and understood will bolster short and long term prediction capabilities.

To adequately cover relevant material, we limit the scope of this paper to social dynamic agents in commonly seen open-ended environments, like homes or roads. We do not cover prediction of general patterns like traffic or weather, nor do we discuss activity-specific ap-

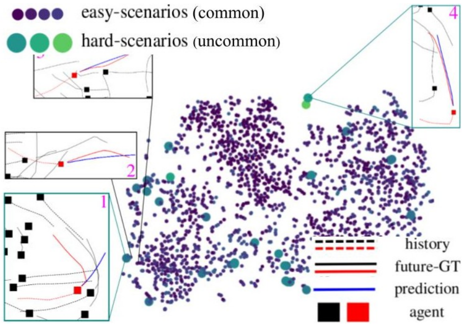

Figure 1: This figure, taken from Makansi et al. (2021), illustrates the confusion within feature space between common and uncommon example embeddings from the UNIV dataset (Pellegrini et al., 2010), with the EWTA model (Makansi et al., 2021) with no contrastive loss, using t-SNE.

plications like cooking or sports. Furthermore, we focus on presenting a variety of methods that differ from each other, and either have made significant contributions to performance, or are currently state-of-the-art methods. Most of these methods involve neural networks.

Our contributions include 1) a comprehensive taxonomy for all types of prediction tasks including tables that show which prediction dataset should be used, given various types of input and environmental constraints, 2) a review of social interaction modeling techniques for prediction, 3) a review of long-tailed learning techniques within classification, regression, and prediction, and 4) a discussion of methodology trends and research directions for prominent tasks. While previous surveys have provided prediction taxonomies (Rasouli, 2020; Gulzar et al., 2021), tables of prediction datasets (Rudenko et al., 2020), and discussed social interaction (Barquero et al., 2022) and long-tailed learning (Zhang et al., 2021b), ours provides a prediction taxonomy that can be applied to any forecasting task, organizes datasets by their modalities and environment characteristics to facilitate dataset choice based on problem/resource constraints, focuses on social interaction techniques in open-ended environments, and to our knowledge, is the first to discuss long-tailed learning in regression and prediction.

1.1 Terminology

In this work, we use the term *dynamic agents* to refer to humans (or autonomous robots that mimic humans) of various classes (pedestrians, cyclists, vehicles, etc.) exhibiting physical movement. Although the categories within our taxonomy could be applied to tasks like predicting the actions of buyers and sellers, investors, or social media followers, these topics do not fall under the category of dynamic agents, and therefore will not be explicitly discussed.

An *open-ended environment* indicates conditions where an agent's future goal or path can take on an unbounded number of possibilities (Min et al., 2014). This puts the focus on environments like roads and malls where people can move in any direction and interact with others in myriad ways, instead of environments like cooking and sports, where people are more likely to follow the physical limits of the location and types of objects used.

By *social interaction*, we refer to interactions between two or more dynamic agents (Barquero et al., 2022) such as one friend following another or redirecting one's path to avoid collision with a cyclist. Since the paths of both agents in such a situation are unknown, agents must exhibit indications of their intended actions to other agents. These indications, as well as the social conventions that guide them, can be modeled and used to improve forecasting. We do not discuss non-social interactions like agent-object or object-object interactions, although they can also be modeled (Tamaru et al., 2021), as an object's future path is typically known.

Lastly, we use the term *long-tailed* to describe most naturally sampled datasets that contain many examples of a few common cases and few examples of many uncommon cases (Makansi et al., 2021), as shown in Figure 8. The uncommon examples in the long tail are harder to predict, as they are rare and dispersed among the many majority cases.

1.2 Methodology

In order to identify relevant papers for the topics discussed within this survey, we input the search terms specified in Table 1 into Google Scholar in July 2022, filtered through the results to find the papers that were relevant to each topic, and followed citations forwards and backwards from all articles relevant to the topics covered to find those which may not have been found by the search. Some of these topics (i.e. those in Section 2) are intended for performance comparisons, and are therefore dataset-specific, while others are general overviews of the topic as a whole. In the rest of this section, we identify the inclusion criteria used to specify which papers are considered relevant.

1.2.1 Inclusion Criteria

For Sections 2.1, 2.2, and 2.3, in order to decide which methods to cover within our performance comparison of methods within each sub-task (i.e. the sub-tasks of RGB Image Generation, Action Anticipation, Early Action Prediction, Trajectory Prediction, and Human Pose Prediction), we 1) identify the most popular dataset used by methods within the sub-task, 2) identify the most popular metrics used by those methods, and 3) filter through the methods which use the most popular dataset and metric to select for 'leading' and/or 'revolutionary' methods (i.e. methods which have the best performance, and/or propose novel contributions which are reproduced or used by many others in the field).

To identify the most popular dataset, we use Fig. 2 of Rasouli (2020), which shows the most popular datasets within dynamic agent prediction and displays the number of papers that use each dataset. We take the singular most popular dataset for each sub-task, based on the popularity count in Rasouli (2020), and filter the search results of each subtask by the corresponding dataset (see the "Suffix" column of sections 2.1, 2.2, and 2.3 in Table 1).

To identify the most popular metrics applied by the methods in each sub-task which use the chosen dataset, we search Google Scholar by the relevant search criteria (each prefix + suffix in Table 1), and perform two searches: 1) sorted by relevance (i.e. highest number of citations) and 2) sorted by date (i.e. most recent, up to one year ago). Then, we view up to 50 search results for each search, record the metrics used by each method that fits our definition of the sub-task (defined in the corresponding sections) and choose the metric which is most popular. We also include relevant methods referenced by Rasouli (2020) within the

Section	Topic	Sort By	No. Results Viewed	Search Criteria Prefix		Suffix
2.1	RGB Image Generation	RD	50	video prediction next frame generation	+	UCF-101
2.2	Action Anticipation	RD	50	action anticipation	+	PIE dataset
	Early Action Prediction	RD	50	action prediction action anticipation early action prediction	+	UCF-101
2.3	Trajectory Prediction	RD	50	trajectory prediction	+	ETH/UCY
	Human Pose Prediction	RD	50	human pose prediction	+	Human3.6M
4	Social Interaction	R	20	-all of the above-	+	social interaction
5	Long-Tailed Learning	R	20	-all of the above-	+	long-tailed imbalanced dataset
3	Datasets	D	20	interaction long tailed imbalanced	+	prediction dataset
1.3	Related Surveys	R	20	-all of the above-	+	prediction survey prediction review

Table 1: Search criteria used on Google Scholar to identify relevant papers for each topic (linked to the section in which that topic is discussed) within this survey. The search criteria in Section 2 are intended for performance comparison of methods within each topic (on a specific dataset) while the search criteria in the other sections are intended for a more general review of the topic. The search criteria for each topic were all combinations of one prefix and one suffix for all prefixes and suffixes specified for that topic. 'All of the above' indicates that the prefixes used for the corresponding topic were the combined set of all prefixes listed in topics placed above that topic within this table. Relevance (R) sorts by number of citations (maximum appears first) and date (D) shows the most recent papers first. No. Results Viewed describes the number of results considered for relevance within each search (each combination of prefix and suffix) for each 'Sort By' type, cut off early if at least 10 consecutive results are irrelevant. Abbreviations: R, relevance; D, date.

set of methods whose metrics are recorded. The reason for performing the second Google Scholar search by date is to allow newer methods which may not have had enough time to be discovered and cited sufficiently, to be identified and surfaced. Additionally, if more than 10 consecutive search results are irrelevant (i.e. do not fit our definition of the sub-task) we stop the search and do not go through the rest of the 50 search results. Afterwards, we perform forwards and backwards searches through the methods found by the Google Scholar search (and from the backwards search of Rasouli, 2020) to identify more relevant methods.

In order to further filter the relevant methods which match our definition of the sub-task and use the identified dataset and metric, we rank the methods by 1) their performance and 2) the number of reproductions and/or comparisons made to it by other methods within our chosen set. Finally, we choose ∼10 methods (depending on how many are available), about half of which showcase methods reproduced/compared to by many others, and the rest of which showcase the highest performing methods. We limit the number of methods discussed to ∼10 in order to clearly represent and compare them in one continuous table.

Social interaction and Long-Tailed Learning. Since social interaction and long-tailed learning are less studied aspects of prediction, we focus on the variety of methods presented instead of on the best performing or most reproduced methods. We search by relevance and stop at 20 results per search because there are so few methods that even by the 10th result we have typically exhausted the number of relevant studies. Furthermore, when we come across multiple studies which use the same social interaction module or long-tailed learning method, we choose the highest performing of the similar methods. For Trajectory Prediction, since it is the only sub-task which yields a plethora of social interaction modeling techniques, we list the different varieties of interaction modules present within the search results (where we also include relevant search results from Section 2.3), and for each type of social interaction module, we choose the highest performing Trajectory Prediction method which employs it.

Datasets. For the datasets presented in Section 3, we mainly focus on the datasets discussed in Rasouli (2020), and describe them in more detail than has been done previously. However, since Rasouli (2020) was published in 2020 and doesn't include many social interaction-specific or long-tailed learning oriented datasets, we supplement this list with datasets found more recently (focusing on those published between 2019 and 2022) by searching Google Scholar with the terms listed in Table 1. Since there are only few such datasets, we stop at 20 results per search.

Related Surveys. In order to identify other related surveys, we use the corresponding search criteria within Table 1, sorted by relevance (since good surveys are more likely to be cited quickly) and viewed the top 20 results for each of the searches performed. From this list, we chose to discuss the surveys which were most similar to one or more of the topics discussed in this work.

1.3 Related Surveys

Dynamic Agent Prediction. There are many existing works that survey the field of dynamic agent prediction. Rasouli (2020) and Hirakawa et al. (2018) both give a broad overview of vision based prediction: Hirakawa et al. (2018) cover probabilistic and deep learning methods in street and crowd navigation, and Rasouli (2020) categorizes the methodologies and provides a comprehensive list of datasets for a range of tasks from future scene generation to trajectory prediction within applications from sports to driving. Georgiou et al. (2018) review trajectory prediction of pedestrians, cars, boats, planes, and animals on datasets primarily involving maps and GPS coordinates while Gulzar et al. (2021) cover both physics and learning based modeling approaches on pedestrians and vehicles across various modalities (map aware, scene aware, interaction aware) and task types (intent prediction, trajectory prediction, occupancy maps). There are also vehicle specific

(Leon & Gavrilescu, 2021; Liu et al., 2021; Paravarzar & Mohammad, 2020) and pedestrian specific (Korbmacher & Tordeux, 2021; Rudenko et al., 2020) surveys. The most notable out of these are Liu et al. (2021) and Rudenko et al. (2020). Liu et al. (2021) provide a succinct table of leading vehicle prediction approaches and their characteristics (input modalities, output modalities, history/scene encoding, interaction encoding), and Rudenko et al. (2020) detail pedestrian prediction metrics, datasets, and methods organised into an insightful taxonomy. Finally, Zhao and Wildes (2021) and Hu et al. (2022) discuss datasets, state-of-the-art performance, and online methods for action prediction.

Social Interaction. Within prediction, Barquero et al. (2022) is the first to combine human motion forecasting and social signal forcasting into one work. Barquero et al. (2022) surveys datasets and methods that study focused social interactions with at least one visual cue, where 'focused' indicates continuous social interaction between agents and includes two-person conversations, but not trajectory prediction. As trajectory prediction is an important area where social interaction modeling is making great strides, we cover social interaction modeling within open-ended environments, and devote attention to trajectory prediction.

Long-Tailed Learning. Though Zhang et al. (2021b) was the first survey to use the term 'long-tailed' to refer to imbalanced datasets, and proposes a novel evaluation metric for the comprehensive list of deep learning methods it categorizes and benchmarks, techniques for countering dataset imbalances predate deep learning. Zhang et al. (2021b), Oksuz et al. (2021), and Krawczyk (2016) focus on techniques developed within computer vision, with Oksuz et al. (2021) studying visual detection and Zhang et al. (2021b) reviewing classification. Meanwhile, other surveys focus on a broad range of applications from cancer detection (Johnson & Khoshgoftaar, 2019) to stock market prediction (Branco et al., 2016), and cover a broad range of fields from management to agriculture (Haixiang et al., 2017). Despite the plethora of research done on long-tailed learning, most studies focus on classification tasks. Ours is the first survey to cover long-tailed learning techniques developed for either regression or prediction.

2. Types of Prediction Tasks

In this section, we describe types of prediction tasks through a taxonomy that categorizes tasks by output format, as shown in Figure 2. Prediction can be performed as a *classification task* with a predetermined set of possible future actions, a *regression task*, where future locations are regressed, or a *generative task*, where the future scene is generated at the pixel level. For each of these categories, we give examples of different types of problem formulations that fall into the category. Although various taxonomies have been defined in other papers like Rasouli, 2020 (where categories consist of 'video', 'action', 'trajectory', 'motion', and 'other' prediction), or Gulzar et al., 2021 (where categories are 'intention', 'unimodal trajectory', multimodal trajectory', and 'occupancy maps'), these taxonomies simply group existing papers into similar types of tasks, and don't provide a framework where any future prediction task can be covered by the taxonomy. For example, the facial pose prediction task studied in Feng et al. (2017) does not fall under any of the categories of either Rasouli (2020) or Gulzar et al. (2021). Our taxonomy, on the other hand, encompasses these taxonomies ('video' and 'occupancy map' prediction are generative tasks,

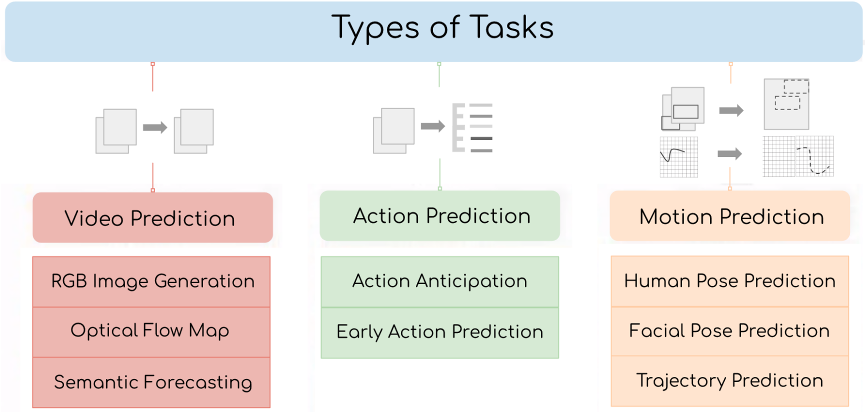

Figure 2: Types of Prediction Tasks

'action' and 'intention' prediction are classification tasks, and 'trajectory' and 'motion' prediction, as defined by Rasouli, 2020, are regression tasks), and can also encompass new tasks like facial pose prediction (which is categorized as a regression task since future locations of facial keypoints are regressed).

However, to make the categories more easily identifiable and consistent with Rasouli (2020), we name each of the three task categories by the content that is being predicted (e.g. Video Prediction, Action Prediction, Motion Prediction) instead of the technical method used to make the prediction (e.g. generative, classification, regression), respectively.

1. **Video Prediction (generative task).** Per-pixel future scene generation in videos, where input and output frames have the same height and width (Rasouli, 2020). The following tasks fall within this category:
   (a) RGB Image Generation - generating future video frames in RGB (Rasouli, 2020)
   (b) Optical Flow Map - generating the optical flow map between the current frame and one or more future frames (Rasouli, 2020)
   (c) Semantic Forecasting - generating the semantic segmentation masks of future scenes from past RGB images
2. **Action Prediction (classification task).** Predicting a future action from a set of action classes (Rasouli, 2020). The following tasks fall within this category:
   (a) Action Anticipation - determining the next action that will follow based on previous actions and other context within the scene (Rasouli, 2020), also called intention prediction in (Gulzar et al., 2021)
   (b) Early Action Prediction - detecting the current action in progress based on partial observation of the action (Rasouli, 2020)
3. **Motion Prediction (regression task).** anticipating the future movements (either by parts or as a whole) of dynamic agents. The following tasks fall within this category:
   (a) Human Pose Prediction - predicting future joint positions or skeletons of agents
   (b) Facial Pose Prediction - predicting future locations of facial landmarks

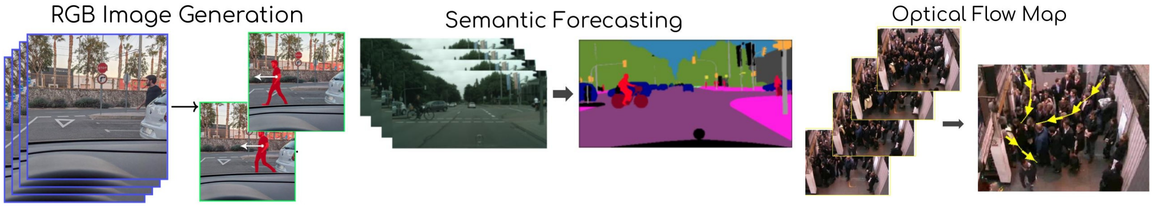

Figure 3: Types of Video Prediction Tasks. Images derived from Oprea et al. (2022) (left and center) and Cheriyadat and Radke (2008) (right) and used for illustration purposes only.

(c) Trajectory Prediction - predicting future locations of dynamic agents on a coordinate system

2.1 Video Prediction

Video prediction is a generative task: whether the scene is represented as an RGB image frame like in (Ying et al., 2019), the optical flow between two frames like in (Zhang et al., 2019), or a birds-eye view of an area where people are coming and going like in (Choi & Dariush, 2019), prediction is done by generating the RGB image, optical flow, semantic segmentation mask, or birds-eye view image that the scene will resemble in the future. Benefits of the RGB Image Generation and Optical Flow Map sub-tasks are that self-supervision can be used to train prediction models using any video dataset, but challenges are that the model needs to perform two difficult functions at the same time: long term dynamic agent forecasting and realistic non-blurry image generation (Zhou et al., 2020). Semantic Forecasting, on the other hand, requires segmentation labeling but can provide more accurate future segmentation masks than can be provided by using RGB Image Generation and then applying semantic segmentation to the generated images (Luc et al., 2017). Another challenge involving supervision of video prediction tasks is that in forecasting, many futures may be equally probable while only one future plays out and gets recorded as ground truth (Mangalam et al., 2020a). Recent works have dealt with this multimodality through probabilistic (e.g., Bhattacharyya et al., 2019) and deep-learning based (e.g., Fugošić et al., 2020; Babaeizadeh et al., 2018; Fragkiadaki et al., 2017) methods.

Datasets. The primary datasets used for evaluating video prediction methods are Caltech Pedestrians (Dollar et al., 2009), KITTI (Geiger et al., 2012), UCF-101 (Soomro et al., 2012), and Human3.6M (Ionescu et al., 2014), and the common metrics used on these datasets are Mean Square Error (MSE) of pixel intensities, Peak Signal-to-Noise Ratio (PSNR), and Structural SIMilarity (SSIM) index. While MSE is used to measure the difference between the ground truth and generated images, PSNR is a metric typically used to evaluate video transmission quality (Chan et al., 2010) and calculated by taking the log of the maximum pixel value in an image (typically 255), divided by the mean squared error of the image (Nasrabadi et al., 2014). SSIM, which is typically used for measuring image quality, is a weighted combination of comparisons between two images in terms of their luminance, contrast, and structure with values ranging between -1 and 1 where 1 is is perfectly similar, 0 is no correlation, and -1 indicates dissimilarity (Wang et al., 2004). Although there are many tasks within video prediction, we focus this section on methods

Model	Input Encoding Method	Modeling Method	Training Method	UCF-101	
				PSNR	SSIM
Liu et al. (2018)	dynamical atoms	sparse motion autoencoder	LTI loss	34.26	0.96
Zhang et al. (2019)	foreground optical flow	autoencoder	style loss+ occlusion hole loss	31.2	0.96
Cho et al. (2021)	CNN	autoencoder+ memory network	L_1+gradient loss	35.5	0.95
Ho et al. (2019)	CNN	sparse motion field	RL	30.8	0.91
Bhattacharjee and Das (2019)	CNN	GAN	feature-matching loss	40.1	0.96
Kwon and Park (2019)	CNN	GAN	adversarial	28.2	0.923
Ying et al. (2019)	CNN	GAN + autoencoder	adversarial	**46.4**	**0.98**
Cai et al. (2018)	skeleton	GAN	feature-matching loss	37	0.86
Liang et al. (2017)	ConvLSTM	GAN	adversarial	30.5	0.94
Byeon et al. (2018)	PMD	PMD	L_1+gradient loss	28.7	0.921
Oliu et al. (2018)	ConvGRU	bGRU	encoder/ decoder skipping	23.872	0.7389

Table 2: Summary of Video Prediction Next Frame RGB Image Generation methodology and performance on UCF-101 (Soomro et al., 2012) dataset using PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural SIMilarity) metrics. Abbreviations: CNN, convolutional neural network; LSTM, long short term memory network; GRU, gated recurrent unit; ConvLSTM, convolutional LSTM; ConvGRU, convolutional GRU; PMD, parallel multi-dimensional unit, which is a type of ConvLSTM; GAN, generative adversarial network; RL, reinforement learning; bGRU, bijective GRU; LTI, linear time-invariant. Bolded numbers indicate best performance and underlined numbers indicate second-best performance (higher is better).

that solve the most commonly studied video prediction sub-task, Next Frame RGB Image Generation, on the most common video prediction dataset, UCF-101 (Soomro et al., 2012).

Input Encoding Methods in video prediction typically consist of CNNs with a U-net (Ronneberger et al., 2015) type architecture (e.g., Cho et al., 2021; Ho et al., 2019; Bhattacharjee & Das, 2019; Kwon & Park, 2019; Ying et al., 2019). CNNs are an obvious choice for encoding the sequences of frames within a video due to their effectiveness on image-based data (Russakovsky et al., 2015), and the U-net architecture is an especially versatile form of CNN which can be applied to any application where the 'labels' are of the same format (e.g. an image with the same height and width) as the inputs (Ronneberger et al., 2015). Since this is the case for video prediction, U-net CNNs can either be used straightforwardly for encoding purposes (e.g., Cho et al., 2021; Ho et al., 2019) or as the generator within a GAN framework (Isola et al., 2018) to perform adversarial learning (e.g., Bhattacharjee & Das, 2019; Kwon & Park, 2019; Ying et al., 2019).

However, these CNNs treat the video simply as a set of frames and ignore their temporal structure. In order to encode temporal structure within the video, some papers employ recurrent methods like convolutional LSTMs, (ConvLSTM) (e.g., Liang et al., 2017), convolutional GRUs, (ConvGRU) (e.g., Oliu et al., 2018), and a new type of ConvLSTM by Byeon et al. (2018) called a parallel multi-dimensional (PMD) unit. ConvLSTMs and ConvGRUs use the memory cells of an LSTM or a GRU, respectively, but they expand the size of the input to include a larger receptive field which shifts across the image like a convolution (Liang et al., 2017). Meanwhile, PMD units apply both convolutional operations and LSTM connectivity to both temporal and spatial dimensions indiscriminately (Byeon et al., 2018). While ConvLSTMs expand the receptive field across the height and width of an image and input one frame at a time into the LSTM, PMDs have 5 parallel units: one does the same as the ConvLSTM, one expands the receptive field across height and time and inputs a slice of width at a time into the LSTM, one inputs a slice of width at a time in the negative width direction, one expands receptive field across width and time and inputs a slice of height at a time into the LSTM, and one inputs a slice of height at a time in the negative height direction. This structure fixes the blind-spot problem of ConvLSTMs (i.e. missing wider spatial context from the more recent timesteps), since ConvLSTMS only aggregate more spatial context as they go further back in time (like a pyramid).

Other types of input encodings which use semantically meaningful ways to restructure and simplify the input include optical flow based separation of foreground from background (e.g., Zhang et al., 2019; Liang et al., 2017), skeleton detection of humans in the image (e.g., Cai et al., 2018), and the use of Linear Time Invariant (LTI) systems to learn a set of suitable poles to use as dynamical atoms, as in (Liu et al., 2018). Optical flow based methods use the difference between two timesteps to find which pixels contain objects which are moving, and calculate the first and second order derivatives of that movement to propagate it into the future (Zhang et al., 2019), while dynamical atoms capture motion by identifying the dynamics of each pixel across time and representing that path as a linear combination of pre-defined dynamical atoms learned during training (Liu et al., 2018). Meanwhile, Cai et al. (2018) first perform skeleton detection on video sequences containing humans, predict how those skeletons will move in the future, and then use the original images to generate an image of the original human in the predicted skeleton position. Each of these methods predicts from a simpler representation of the video (e.g. flow, dynamical atoms, skeletons) by capturing a key aspect of the video (e.g. movement, human pose), and finds that doing prediction on these simpler, lower-dimensional, and more meaningful representations helps to better predict the entire future frame.

Modeling Methods. While most of the reviewed methods use generative adversarial networks (GANs) (e.g., Bhattacharjee & Das, 2019; Kwon & Park, 2019; Ying et al., 2019; Cai et al., 2018; Liang et al., 2017), or variations of autoencoders (e.g., Cho et al., 2021; Zhang et al., 2019; Oliu et al., 2018; Liu et al., 2018), Byeon et al. (2018) use 4 layers of PMD units with 2 skip connections, and Ho et al. (2019) identifies the motion vectors of only the critical pixels of the image and uses that to estimate the future motion vectors of those pixels to generate future frames. Autoencoders are a natural framework to use in video prediction because they aggregate both the global context necessary to understand which parts of the scene are moving and why, and local context which helps generate the pixel-by-pixel texture of the future frame (Ronneberger et al., 2015). In addition to having an autoencoder, Cho et al. (2021) uses a memory network on the scene encoding which memorizes the prototypical motion patterns of different types of objects in order to increase robustness to scenes in different environments. Furthermore, both Cho et al. (2021) and Zhang et al. (2019) add skip connections to the autoencoder in order to add local context (as shown in Ronneberger et al., 2015), and Oliu et al. (2018) uses a bijective gated recurrent unit (bGRU) within the autoencoder to allow shared states between the encoder and decoder. In Liu et al. (2018), the dynamical atoms from the input encoding are used by the decoder to predict the next frame by extending their temporal horizon into the future. GANs, on the other hand, are good at performing the realistic image generation part of the video prediction problem (Bhattacharjee & Das, 2019). While Bhattacharjee and Das (2019) implement a straightforward GAN, Ying et al. (2019) and Liang et al. (2017) implement two different generators: an image generator which directly predicts the future frame and an optical flow generator that predicts the difference between the last frame and the predicted frame. However, Ying et al. (2019) combines their outputs using an autoencoder network, while Liang et al. (2017) apply two different discriminators, one for the images predicted by the two generators (one directly predicted and one whose optical flow has been applied to the last frame), and one for the predicted optical flows of the generators. Cai et al. (2018), on the other hand, employs two whole GANs and an autoencoder: it uses the first GAN to estimate the pose of the skeleton within the history sequence, the second GAN to predict the future poses, and inputs the future predicted pose and original image into an autoencoder (with skip connections) to generate the future frame. Meanwhile, in order to decrease the likelihood of blurry predictions for predicted frames that are farther into the future, the generator of Kwon and Park (2019) first predicts the future frame using history frames, then tries to predict the first frame using all the other frames (including the predicted future frame). To enforce temporal consistency, they also employ two discriminators: one for the generated images and one for the entire sequence.

Training Methods. Although novel systems like Liu et al. (2018) use LTI-specific training methods where the poles and regularizing parameters get updated by a standard L_2 loss, most methods add visual realism terms to the loss function to increase the quality of the predicted image. For example, Zhang et al. (2019) determines whether there are holes in the generated frame due to occlusions and adds this measure to the loss function while Cho et al. (2021), Byeon et al. (2018) use differences in spatial gradients across the output image to force the loss function to preserve edges and reduce blurriness. Cho et al. (2021) also uses L_1 instead of L_2 norm to further reduce blurriness in generated frames. While most GANs use a simple adversarial loss (e.g., Kwon & Park, 2019; Liang et al., 2017; Ying et al., 2019), Cai et al. (2018) and Bhattacharjee and Das (2019) also add terms that enforce details and reduce blurriness by adding a feature-matching loss which compares the activations in a pre-trained visual perception network that is applied to the ground truth image and the generated image respectively and tries to reduce that difference. Finally, Ho et al. (2019) uses a reinforcement learning framework to iteratively generate future frames from the sparse motion field (composed of a few critical pixels and their estimated motion vectors) for increased efficiency, and Oliu et al. (2018) alternately updates encoder and decoder weights to prevent the network from learning an identity model.

Discussion. A summary of the described methods and their performances can be seen in Table 2. It can be observed based on the trends present within this selection that the more straightforward methods with simpler input encoding and training schemes seem to perform better. For example, compared to complicated architectures like the PMD, bGRU and forwards/backwards generators (Kwon & Park, 2019), all of which make fundamental changes to the internal structure of the basic LSTM or GAN, methods which use a simple GAN (Bhattacharjee & Das, 2019), stack multiple simple GANs (Cai et al., 2018), or use one standard discriminator (Ying et al., 2019) instead of multiple with different purposes

Model	Agent Encoding	Scene Encoding	Modeling	Training	PIE	
	Method	Method	Method	Method	Accuracy	F1
Lorenzo et al. (2021b)	bbox Transformer	body crops+ Transformer	MLP	re-weighting	-	0.779
Lorenzo et al. (2021a)	bbox+skeleton Transformer	body crops+ RubiksNet	MLP	re-weighting	0.89	0.81
Gesnouin et al. (2021)	bbox+skeleton GRU+CBAM	-	attention	re-weighting	0.88	0.80
Yao et al. (2021)	bbox FCN	CNN+ARN	MLP	multitask loss + re-weighting	0.82	**0.90**
Chen et al. (2021)	bbox+skeleton	CNN+GCN	LSTM	cross-entropy loss	0.79	0.78
Rasouli et al. (2021)	bbox + occupancy grid LSTM	semantic segmentation	LSTM	multitask loss + re-weighting	**0.91**	0.85

Table 3: **Action Prediction**: Summary of **Action Anticipation** methodology and performance on the PIE (Rasouli et al., 2019) dataset using classification Accuracy and F1 score (function that summarizes precision and recall) metrics. Dashes in metrics indicate that this metric was not included in the paper. Abbreviations are: ARN, attentive relation network; CBAM, convolutional block attention module; bbox, bounding box; GCN, graph convolutional network; FCN, fully connected network; RubiksNet, spatiotemporal shift based alternative to 3D CNN (Fan et al., 2020). Bolded numbers indicate best performance and underlined numbers indicate second-best performance.

(Liang et al., 2017) seem to yield better performance on next frame image generation, possibly because they may be easier to train. One exception, however, is the addition of feature-matching loss (Bhattacharjee & Das, 2019; Cai et al., 2018) which does seem to improve performance, probably due to its effect on the clarity of the output image

Moreover, use of autoencoders and GANs seem to outperform recurrent networks across the board. Although recurrent networks seem like they could be useful to capture the history of an agent, their performance is lower, possibly due to two reasons: next frame prediction may not be far enough in the future to require involved time series modeling, and metrics may optimize for the realistic-looking image generation aspect (which GANs explicitly aim for) more strongly than the dynamic agent forecasting aspect (which relies more on the history encoded by recurrent networks). Finally, one interesting observation is that Ying et al. (2019), the highest performing method of those selected, and Kwon and Park (2019), the lowest performing method of those selected, seem to have very similar methodology (the main difference is that Kwon & Park, 2019 use forwards and backwards predictions to boost performance while Ying et al., 2019 generate image and optical flow predictions). Furthermore, Ying et al. (2019) and Liang et al. (2017), another method performing on the poorer end of the spectrum, are also very similar in that they both use GANs to generate both image and optical flow predictions. Therefore, the biggest strength of this method (Ying et al., 2019) may possibly lie in its one unique aspect among this selection: the use of both GANs and autoencoders, the two best performing modeling methods.

2.2 Action Prediction

Action prediction is a classification task where the history, typically given as a video of the scene, is processed to predict a future or ongoing action. Since it's a classification task, many techniques from image classification and video recognition have been adapted to action prediction. The two main sub-tasks that fall under this category are action anticipation (predicting the future action without seeing any of the action in the history), and early action prediction (prediction based on partial observation of the action) (Rasouli, 2020). In action prediction, the classification scores of the other action classes can indicate multimodality based on how close they are to that of the predicted future action (Richter et al., 2022). However, the limited nature of classification means that unforseen actions cannot easily be identified, which poses a problem in open-ended environments where unexpected actions are of interest (Rosenfeld & Ullman, 2018).

2.2.1 Action Anticipation

In this section, we discuss the most recent prominent methods in action anticipation by comparing methods which perform road-crossing anticipation (the task of determining whether or not a pedestrian intends to cross the road in the next few seconds). As detailed here and summarized in Table 3, we compare their input encoding, modeling, and training methods, as well as their performance.

Datasets. Although there are many studies of action anticipation, most of these use datasets in environments like kitchens and warehouse assembly, which is out of the scope of this work, as they are not open-ended environments. The primary datasets currently used for action anticipation in open-ended environments are BDD100K (Yu et al., 2020), JAAD (Kotseruba et al., 2020), and PIE (Rasouli et al., 2019), and to our knowledge, there are fewer works evaluated on these datasets than those in other subtasks because most of the methods within this field focus on more controlled environments. The methods discussed in this section are primarily evaluated on the most popular of these datasets, the PIE dataset (Rasouli et al., 2019), which includes driving scenes recorded from a moving vehicle (i.e. image frames and vehicle kinematic information), per-frame bounding boxes of the agent whose future is to be predicted, and human annotated labels for crossing intention prediction (in which human labelers guess whether or not the pedestrian intends to cross the road). The primary metrics used within action anticipation are Accuracy (number of correct predictions divided by total number of predictions), mAP (mean average precision), F1 score (function that summarizes precision and recall), and AUC (area under precision-recall curve). The methods in Table 3 measure performance primarily by Accuracy and F1 score.

Agent and Scene Encoding Methods use the three inputs provided (images, ego vehicle kinematics, and bounding box coordinates of each agent for every timestep) as the input modalities representing the agent and/or the scene, and then encode them.

The most popular agent history encoding method is to take the bounding box coordinates of the pedestrian whose crossing intention is to be predicted, as well as the image of the frame it belongs to, perform human pose detection on the crop to extract the skeleton of the pedestrian, and use both the sequence of skeletons and sequence of bounding box coordinates as inputs to the agent encoder (Lorenzo et al., 2021a; Gesnouin et al., 2021;

Chen et al., 2021). Since a pedestrian's pose is a key indicator of intention to cross (e.g. looking towards oncoming cars, leaning forward, etc.) using pose skeletons as part of the input makes a big difference in being able to detect crossing intention (Gesnouin et al., 2021). A few methods (e.g., Lorenzo et al., 2021a, 2021b; Gesnouin et al., 2021; Rasouli et al., 2021) also input ego vehicle kinematics directly into the agent encoder network, and Lorenzo et al. (2021a) find that this input modality significantly improves performance, potentially due to the fact that pedestrians react differently to vehicles moving at different speeds. While Lorenzo et al. (2021a, 2021b) use a Transformer architecture to encode the bounding box, skeleton, and ego motion in order to learn the relationships between all samples in the input sequence (Lorenzo et al., 2021a), Chen et al. (2021) simply concatenate the bounding box and skeleton to input into the decoding stage of the prediction and Gesnouin et al. (2021) use atrous convolutional spatiotemporal attention modules (CBAMs) to encode the sequence of skeletons and separate GRUs (to encode skeleton information, bounding boxes, and ego motion), which feed into a Temporal attention network. Each skeleton sequence gets processed by 3 CBAM branches with different dilation factors to allow the network to directly work at different time resolutions, while the GRUs allow the model to exploit long-term temporal patterns on the location-invariant inputs. Instead of using skeletons, Yao et al. (2021) simply applies a fully connected layer to the bounding boxes and Rasouli et al. (2021) encodes 3 input modalities (sequence of bounding boxes, ego vehicle kinematics, and an occupancy grid that indicates where on the map the pedestrian has traveled) using 3 separate LSTMs to generate an independent representation for each modality.

Although Gesnouin et al. (2021) eschews using scene information for being too "sensitive to noise, background, and illumination conditions," other methods incorporate information about the scene either through body crops (cropping the image frame around the pedestrian whose future is being predicted, to the size of his/her bounding box), graph/relational networks (which aim to encapsulate the relationship between all objects and agents in a frame), and semantic segmentation of the entire frame. These methods then have a sequence of body crops, graphs, or segmented frames for each timestep. To encode the body crops, Lorenzo et al. (2021b) uses a spatiotemporal-attention based Transformer network while Lorenzo et al. (2021a) employs RubiksNet (Fan et al., 2020), a method that learns temporal, vertical, and horizontal shift parameters as an alternative to 3D convolutional networks. Both of these methods attempt to record spatiotemporal information within the sequence of body crops in different ways. Meanwhile, Yao et al. (2021), Chen et al. (2021), and Rasouli et al. (2021) attempt to encode the different objects and agents in the scenes and their relations to each other using ARNs (attentive relation networks), GCNs (graph convolutional networks), and semantic segmentation, respectively. While the ARN uses a simple soft attention module on top of a FCN to identify traffic neighbors, traffic lights, traffic signs, crosswalks, and bus/train stops (as well as an additional input of ego vehicle kinematics), the GCN extracts visual features of every object and agent in each frame using a pre-trained classification algorithm and connects them into a compact graph using graph convolutions and a graph autoencoder. Since pedestrians' behaviors are often influenced by what is around them, these methods add useful relational scene information into the model (Rasouli et al., 2021).

Modeling and Training Methods. While most methods use a simple MLP (e.g., Lorenzo et al., 2021b, 2021a; Yao et al., 2021) or a simple attention module to combine the input modalities (e.g., Gesnouin et al., 2021), some use LSTM layers for the encoder and decoder (e.g., Chen et al., 2021; Rasouli et al., 2021). While the attention module has the advantage of learning the dependencies among the different modalities (e.g., Gesnouin et al., 2021), the LSTM layers are able to explicitly temporal dependencies within the scene history sequences and agent history sequences both individually and when combined (Chen et al., 2021).

For training, most methods seem to use re-weighting as a class balancing method in order to counteract biases in the model due to imbalanced datasets (see Section 5 for more details) (e.g., Lorenzo et al., 2021b, 2021a; Gesnouin et al., 2021; Yao et al., 2021; Rasouli et al., 2021). Additionally, some use a multitask loss, either for joint action detection and action prediction (e.g., Yao et al., 2021) or joint action prediction and trajectory prediction (e.g., Rasouli et al., 2021). This multitask learning framework induces the model to learn more representative features by adding semantically related learning objectives (Rasouli et al., 2021).

Discussion. Within the agent encoding methods shown in Table 3, we see that most of the methods detect and use pedestrian skeletons as an input modality. However, this does not seem to necessarily lead to higher performance, as the two highest performing methods on each metric, Yao et al. (2021) and Rasouli et al. (2021) do not use pedestrian pose information. Furthermore, the multiple in-depth ablation studies done in Lorenzo et al. (2021b) lend support to the hypothesis that adding a pose modality actually decreases the performance, but more work needs to be done to understand why. Moreover, due to the fact that the lowest performing method is the only method which doesn't use re-weighting (in addition to the information about re-weighting methods presented in Section 5 and supported by Zhang et al., 2021b), it can be concluded that the PIE dataset is relatively imbalanced and that re-weighting methods are necessary when working with this dataset. Finally, the models that use attention for agent encoding or modeling (e.g. Lorenzo et al., 2021a; Gesnouin et al., 2021; Yao et al., 2021), but not for scene encoding like Lorenzo et al. (2021b), seem to be performing on the higher end of the spectrum, indicating that attention may be a promising direction to follow, but more work needs to be done to analyze the role of attention and where it should be applied.

2.2.2 Early Action Prediction

In this section we present a selection of high performing early action prediction methods, chosen to maximize the variety of the techniques represented. These methods attempt to predict a variety of actions across different environments and contexts using the UCF-101 (Soomro et al., 2012) dataset.

Datasets. The most commonly used early action prediction dataset is UCF-101 (Soomro et al., 2012), but other interaction-oriented datasets like UT-Interaction (Perez et al., 2021) and BIT (Kong et al., 2012) are also commonly used. Table 4 summarizes a selection of early action prediction methods and performances evaluated on UCF-101. The most commonly used metric is Accuracy (number of correct predictions divided by total number of

Model	Input Encoding Method	Modeling Method	Training Method	UCF-101	
				20%	50%
Cho and Foroosh (2018)	CNN	Codebook T-CNN	L_2 cross entropy	86.7	90.1
Tao et al. (2021)	BN-Inception	Codebook GAN	adversarial	**94.6**	96.45
Devarakonda and Mukherjee (2021)	3D CNN	MDP	reinforcement learning	85.1	90.5
Wu et al. (2021)	Resnet50+GGNN +attention	LST-GCN	graph alignment loss	88.5	90.9
Shi et al. (2018)	InceptionV3	LSTM-GAN	adversarial	-	98.0
Chen et al. (2022)	skeleton-LSTM	GAN	adversarial	-	94.0
Gammulle et al. (2019)	Resnet50+LSTM +attention	GAN	adversarial	84.2	**98.9**
Wang et al. (2019)	3D CNN	LSTM/Bi-LSTM	teacher-student	87.13	90.85
Liu et al. (2021)	BN-Inception	LSTM/Bi-LSTM	teacher-student	-	93.59
Tran et al. (2021)	TSM	GRU/Bi-GRU	teacher-student	90.5	-

Table 4: **Action Prediction**: Summary of **Early Action Pediction** methodology and performance on the UCF-101 (Soomro et al., 2012) dataset using percentage Accuracy metrics, in observation ratios of 20% and 50%. Dashes in performance indicate that this metric was not included in the paper. Abbreviations are: 3D CNN, 3-dimensional convolutional neural network; LST-GCN, long short term graph convolutional network; K-NN, k-nearest neighbors; T-CNN, temporal CNN; GGNN, gated graph neural network; MDP, markov decision process; TSM, temporal shift model (Lin et al., 2019); GNN, graph neural network; bi-LSTM/GRU, bidirectional LSTM/GRU; Resnet50/BN-Inception/InceptionV3, CNN architectures from He et al. (2015), Ioffe and Szegedy (2015) and Szegedy et al. (2015) respectively. Bolded numbers indicate best performance and underlined numbers indicate second-best performance.

predictions) where some portion (e.g. 10%, 20%, 50%) of the action, timewise, is shown to the model so that the action category can be predicted.

Input Encoding Methods. Although most methods encode the spatiotemporal information present in videos through various types of two-stream CNNs, where a spatial stream processes the video frames one image at a time and the temporal stream processes the optical flow between the frames one optical flow at a time (e.g., Cho & Foroosh, 2018; Tao et al., 2021; Gammulle et al., 2019; Liu et al., 2021; Wu et al., 2021), some methods treat the video as a 3D structure with two spatial dimensions and one temporal dimension on which to perform 3D convolutions (or 3D convolution alternatives such as temporal shift modeling) such as those of Devarakonda and Mukherjee (2021), Wang et al. (2019), Tran et al. (2021). Still others, like Chen et al. (2022), use an LSTM on the detected skeletons of agents, while Wu et al. (2021) create spatiotemporal scene graphs between objects and frames in the scene using gated graph neural networks.

Shi et al. (2018), however, only use per-frame spatial information when performing input encoding (via the Inception V3 framework from Szegedy et al., 2015), and only take temporal information into account during the prediction. This is because the prediction is done by iteratively generating the feature vectors of the next frame using a recurrent network, and classifying the action once 100% of the frames have been generated. The

Inception V3 architecture was chosen for its performance after comparing different CNNs for feature extraction in an ablation study.

Of the two-stream CNNs, some methods combine the outputs of the spatial and temporal CNNs at the frame level (e.g. Cho & Foroosh, 2018; Wu et al., 2021 combine the per-frame image and optical flow feature vectors using an FCN and process each frame separately within the model) while others (e.g. Tao et al., 2021; Gammulle et al., 2019) use separate spatial and temporal processing streams. Tao et al. (2021) use the pre-trained BN-Inception CNN from Wang et al. (2016) to process each frame (and optical flow), combine them across time into two separate processing streams (an image and an optical flow stream) using two FCNs, and eventually make two separate predictions that are only fused at the end, while Gammulle et al. (2019) puts each stream through a pre-trained Resnet50 CNN, an LSTM network, and then a weighted attention layer before concatenating the streams into a context descriptor. These methods use pre-trained models due to their robustness to background and illumination changes and tendency to capture the overall meaning of the input frame (Gammulle et al., 2019), and use two separate processing streams because they naturally provide complimentary information: the image stream captures the objects observed while the optical flow stream captures their movement (Tao et al., 2021). Meanwhile, Cho and Foroosh (2018) processes each frame separately and treats the sequence of frames as words in a sentence in order to leverage temporal prediction concepts from natural language processing. Liu et al. (2021) also use BN-Inception on optical flow frames, but hypothesize that high sampling rates introduce more irrelevant information, which causes misclassification. So instead of processing every frame, they sample only a few frames from the original video and perform an ablation study measuring the influence of different sampling rates on performance, finding 15-20 samples to yield the best results.

Chen et al. (2022) and Wu et al. (2021), however, try to reduce the dimensionality of the input video in different, more semantically meaningful ways. Chen et al. (2022) reduces each frame to the set of extracted skeletons representing the poses of each agent in each frame of the video. These skeletons are then input into an LSTM network to gather temporal context, and treated as probabilistic distributions to model the uncertainty inherent in using partial sequences (Chen et al., 2022). Wu et al. (2021), on the other hand, creates a relational graph between all objects and agents in each frame of the video and connects them across time. To do this, they first use a pre-trained Faster R-CNN model to extract the bounding boxes of all objects and agents in the image, and then run a pre-trained Resnet50 model on various image crops and their optical flows to create a spatial graph where the nodes contain the feature vectors of cropped out objects/agents while the edges contain the feature vectors of the bounding box union between two objects/agents. Then, a gated graph neural network (GGNN) is applied to the spatial graph along with a soft attention mechanism to aggregate the different features. Finally, temporal information is encoded by forming a scene graph where each node is the spatial graph of one frame, and directed edges encode relations between different frames. While the Resnet50 model is chosen via an ablation study of various CNNs, the entire framework is intended to mimic the human tendency to anticipate the propagation of visual relationships in unobserved parts of partial videos (Wu et al., 2021).

Finally, in order to extract spatiotemporal features without having to rely on the limited training data in UCF-101 (Wang et al., 2019), many methods use 3D convolutional (or convolution-style) methods pretrained on the Kinetics action classification dataset (Kay et al., 2017). Devarakonda and Mukherjee (2021) and Wang et al. (2019) use a pre-trained 3D ResNext (Hara et al., 2018) model while Tran et al. (2021) use a TSM (temporal shift module) network (Lin et al., 2019) which uses 3D shift operations as a more efficient alternative to convolutions.

Modeling and Training Methods. Based on the intuition that a longer action sequence is less ambiguous than a shorter action sequence (Tran et al., 2021), many methods (e.g. Wang et al., 2019; Liu et al., 2021; Tran et al., 2021) use knowledge distillation techniques like teacher-student methods, where a teacher model is trained to recognize actions from full videos and a student model is trained to predict early actions from partial videos. (Wang et al., 2019). In order to leverage action recognition methods (which predict action class using the full video), these methods use a bidirectional LSTM or GRU, which is widely used in action recognition, for the teacher model and a single directional LSTM or GRU for the student model. One advantage of this setup is that bidirectional recurrent networks can provide a latent feature representation of the videos at any progress level (Wang et al., 2019), which allows us to force the latent feature representations within the student model to be consistent with those of the teacher model at each progress level. In addition to the loss between the latent feature representations, these methods also force the student network to output the same probability vector produced by the teacher network, instead of the binary label vector which trains the teacher network (Tran et al., 2021).

Due to studies showing that humans build a mental image of the future before initiating motor controls, some methods capture information about the expected future by generating future scene representations (e.g. future skeletons, RGB images, optical flows, or simply future feature vectors), and using a discriminator in a GAN structure to make these expected future representations more accurate (e.g. Chen et al., 2022; Gammulle et al., 2019; Shi et al., 2018; Tao et al., 2021). Chen et al. (2022) use the skeleton sequence embeddings of each agent in the partial video to 1) preliminarily predict an action class for each agent, 2) generate the future skeleton representations of each agent based on its predicted class, and 3) apply a predictor that's pre-trained on full-length sequences to the entire sequence of skeletons (input + generated) of all agents. The loss function includes the per-agent preliminary prediction error, final prediction error, and an adversarial loss from the discriminator which decides whether the generated skeleton sequence is real. Meanwhile, Shi et al. (2018) use multiple LSTMs on the feature vector of each frame to predict the feature vector of the next frame while the discriminator judges whether those feature vectors are real or synthetic. During inference, the full video is iteratively generated by taking the last frame of the partial video as the first input and using the predicted frames as inputs to generate subsequent frames until 100% of the video is generated. Then, this partially generated video is classified using an action recognition CNN. This network is further regularized by allowing the multiple LSTMs which act on different areas of the image to share weights. Meanwhile, Gammulle et al. (2019) use a classification CNN directly on the context feature described in the input encoding section above, but they train this context feature by using it in two GANs: one which generates future RGB frames and one which generates future optical flows. Each GAN also has its own discriminator which attempts to differentiate between the generated future frames (or optical flows) and the ground truth, and in addition to the adversarial and classification loss terms, a temporal regularisation mechanism is applied which compares the predicted embeddings with the ground truth future embeddings. Tao et al. (2021) also employs two GANS (one spatial and one temporal) but to generate full videos, they use a codebook style key-value memory network architecture where similar partial videos are captured within each key memory slot and their corresponding full videos are captured by the value memory slots through training. The advantage of using this key-value structure lies in separating the learning process for different purposes: the key memory matrix can focus on memorizing different types of partial videos and the value memory matrix is trained to distill useful information from full videos for generation (Tao et al., 2021). For each stream (temporal and spatial) of their network, the partial video encoding vector updates key memory slots which have similar encoding vectors, followed by soft attention over all slots, while the value memory slots are updated via a gate mechanism with attention in order to dynamically update or forget full videos attended to by different queries. Finally, the discriminators add adversarial loss by differentiating between the generated and real videos or optical flows.

Cho and Foroosh (2018) also use codebook style learning, but without a GAN: they map the input encoding of each frame to a codeword, arrange the codewords side-by-side in time order, and then apply a CNN to the array of codewords. This method is inspired by techniques developed to perform sentence classification in natural language processing (NLP), and the typical L_2 cross entropy loss is used for training. On the other hand, Devarakonda and Mukherjee (2021) use reinforcement learning to train a Markov Decision Process (MDP) where the video is split into 10 equal segments, each of which is considered a state, and an action is predicted iteratively for each state, with a reward given when the correct class is predicted. This framework was chosen because of its success in modeling tasks like playing Atari (Mnih et al., 2013), since such tasks have parallels to action prediction in deciding future movements. Finally, Wu et al. (2021) apply a short long term graph convolutional network (LST-GCN) to the relational graph of the partial video, which enables the relational graph to update each node and edge based on its spatio-temporal neighbors (using spectral convolutional networks) across various timescales, with attention applied between similar nodes. To predict the future, a structured graph autoencoder is applied to this graph at different timescales to learn spatio-temporal relations of the full video while preserving the temporal relations and spatial features learned from the partial video, and this is concatenated with the graph output from the LST-GCN to perform classification via a soft attention mechanism that aggregates the graphs' features. When training, the LST-GCN is also applied to the full ground-truth video and unobserved part (separately) and two graph alignment losses are used to measure the difference between the generated future graph and ground-truth graph of 1) the full video and 2) the unobserved part of the video.

Discussion. As can be seen by the performances of each of these methods in Table 4, the most apparent trend is the better performance of methods which utilize adversarial learning. Almost half of the methods presented use GANs, and out of these, three-fourths are the highest or second highest performing methods by at least one of the two presented metrics. Furthermore, through the ablation study done in Gammulle et al. (2019), it can be seen that simply adding a generator (which models future visual representation) to a classification model without adversarial learning, can improve performance. This shows

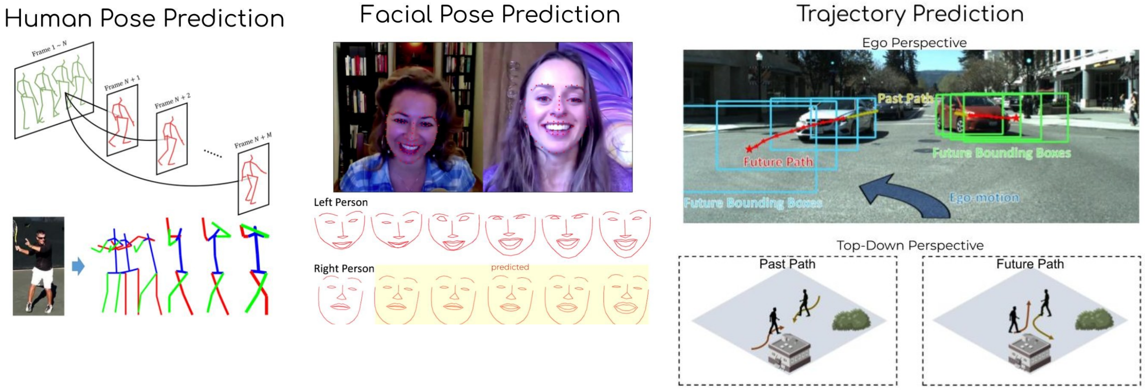

Figure 4: Types of Motion Prediction Tasks. Images derived from Li et al. (2021a) (top left), Chao et al. (2017) (bottom left), Feng et al. (2017) (center), Yao et al. (2019) (top right), and Dwivedi et al. (2020) (bottom right).

that the human tendency of 'building a mental image of the future first' may be effective for neural networks as well.

Another interesting observation is that both of the highest performing methods at the 20% and 50% ratios, Tao et al. (2021) and Gammulle et al. (2019) respectively, use separate spatial and temporal processing streams to encode the input. This could be because this structure resembles the way that human brains process visual input in (initially) relatively separate temporal and spatial pathways (Gundavarapu et al., 2019). Meanwhile, most input encoding methods which treat each frame separately (e.g. Cho & Foroosh, 2018; Wu et al., 2021; Liu et al., 2021), put the frames through a pre-trained 3D CNN (e.g. Devarakonda & Mukherjee, 2021; Wang et al., 2019), or reduce the dimensionality of the input before encoding (e.g. Wu et al., 2021; Chen et al., 2022; Liu et al., 2021) seem to have relatively lower performance. However, there are notable exceptions like Shi et al. (2018), which processes frame-by-frame spatial-only feature vectors into an LSTM and Tran et al. (2021), which uses a pre-trained TSM, which still have very high performance. Therefore, more work must be done to compare different input encoding methods through an ablation study. For Wang et al. (2019) and Liu et al. (2021) however, every aspect of the methods are identical except for the input encoding, which allows us to conclude that reducing the dimensionality of the input through a lower sampling rate seems to perform better than using a pre-trained 3D CNN.

One notable aspect of the two metrics presented is that some methods perform very well on one metric but poorly on the other (e.g. Wu et al., 2021; Wang et al., 2019; Gammulle et al., 2019), likely because they focus too much on one specific timescale and not enough on larger or smaller ones (Wu et al., 2021). The exception, however, is Tao et al. (2021) which performs exceptionally well on both metrics.

2.3 Motion Prediction

Motion prediction is a regression task, where a series of coordinates that correspond to an agent's future pose or location are predicted using their past pose or location, sometimes

Model	Input Encoding Method	Modeling Method	Training Method	Human 3.6M Walking (MAnE)		
				320 (ms)	560 (ms)	1000 (ms)
Mao et al. (2019)	DCT	GCN	L_1 joint angle error	0.49	0.65	0.67
Mao et al. (2020)	DCT + temporal attention	GCN	L_1 joint angle error	0.46	0.59	0.64
Cao et al. (2022)	DCT + dual attention	GCN	L_1 joint angle error	**0.44**	$\underline{0.57}$	$\underline{0.62}$
Li et al. (2021a)	GCN-TCN + DCT	GCN-TCN	multitask action classification loss	$\underline{0.45}$	**0.54**	**0.5**
Wang et al. (2019)	GRU	MDP (GAIL)	adversarial	0.53	0.67	0.69
Gui et al. (2018)	velocity + class + GRU	GRU GAN	adversarial	0.63	-	0.91
Gopalakrishnan et al. (2019)	velocity + class	bi-GRU	dynamic closed + open loop loss	0.64	1.026	1.231
Jain et al. (2016)	spatiotemporal graph	graph LSTM	noise regularization	1.6	1.9	2.13

Table 5: **Motion Prediction.** Summary of **Human Motion Prediction** methodology and performance on the Human 3.6M (Ionescu et al., 2014) dataset's 'Walking' class examples, measured in MAnE (mean angle error) of predictions that forecast 320, 560, and 1000 milliseconds into the future. Dashes in the performance indicate that this metric was not included in the paper. Abbreviations are: DCT, discrete cosine transform; GCN, graph convolutional network; TCN, temporal convolutional network; MDP, markov decision process; GAIL, generative adversarial imitation learning; IRL, inverse reinforcement learning; NN, recurrent neural network; bi-GRU, bidirectional GRU; class, action class label; velocity, velocity of joint angles. Bolded numbers indicate best performance and underlined numbers indicate second-best performance.

in combination with other features like ego video (e.g., Adeli et al., 2021), maps (e.g., Salzmann et al., 2021), head orientation (e.g., Haddad & Lam, 2021), body positioning (e.g., Wang et al., 2021), GPS location, (e.g., Sadeghian et al., 2018b), and/or extracted visual features from cropped images (e.g., Haddad & Lam, 2021) of the agents in the scene. Multimodality in motion prediction is a large area of interest in both trajectory (e.g., Dong et al., 2021; Kosaraju et al., 2019; Gu et al., 2022) and pose (e.g., Fragkiadaki et al., 2017; Gu et al., 2021; Yan et al., 2018) prediction. Although multimodality is still a new research area in pose prediction, it has been widely studied in trajectory prediction using both probabilistic methods like (conditional) variational auto encoders (VAEs or CVAEs) (e.g., Zhou et al., 2021; Xu et al., 2022) and deep neural net training techniques (e.g., Makansi et al., 2019), and is a topic of interest in many trajectory prediction surveys like Gulzar et al. (2021), Rudenko et al. (2020) and Korbmacher and Tordeux (2021).

2.3.1 Human Pose Prediction

In this section, we discuss the prominent human motion prediction methods shown in Table 5 and outline their input encoding, modeling, and training methods. Then, we compare their performance on a popular dataset and metric (also shown in Table 5) and discuss the links which can be made between the input encoding, modeling, and training methodologies, and their performance.

Datasets. In human pose prediction datasets, history and future poses are typically represented either as a set of joints (e.g., Ionescu et al., 2014), a mesh (e.g., von Marcard et al., 2018), or a skeleton (e.g., Yuan & Kitani, 2019) as shown in Figure 5. In this

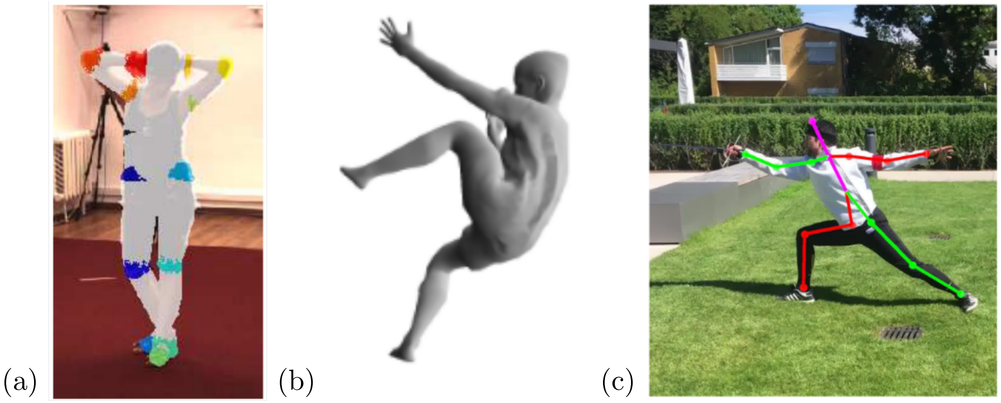

Figure 5: Types of input representations in human pose prediction tasks: (a) set of joints, (b) mesh representation, (c) skeleton representation, with (a) derived from Ionescu et al. (2014); (b) and (c) from von Marcard et al. (2018).

section, we focus on a few prominent human motion prediction methods evaluated on the Human3.6M (Ionescu et al., 2014) dataset, the most popular human motion prediction dataset, which includes both joint and mesh labels, and action class labels for each video segment. In Table 5, we compare the best performing models on the 'walking' class examples of the dataset at timesteps 0.32, 0.56, and 1 second into the future using the MAnE (mean angle error) metric, as this is the most popular way in which current human motion prediction methods are evaluated.

Input Encoding Methods. One of the most common input encoding methods is using a discrete cosine transform, or DCT (e.g., Cao et al., 2022; Mao et al., 2019, 2020; Li et al., 2021a), which uses a linear combination of sinusoidal functions of different frequencies to represent joint positions, in order to explicitly inject temporal information into the encoding (Li et al., 2021a). First proposed by Mao et al. (2019), this technique offers two main advantages: it yields a continuous vector which contains more information than a one-hot embedding of the position, and the resulting positional embedding is highly correlated, making the embeddings of poses at successive timesteps are more similar (Li et al., 2021a). Furthermore, the higher end DCT values can be ignored in order to remove high motion frequencies and provide a compact representation that captures smoothed human motion (Mao et al., 2019). In methods that use DCT, future prediction is also typically done in DCT space and then converted back to angular space or 3D joint positions (Mao et al., 2019). Some methods also add other input encoding methods on top of the DCT, like attention (e.g., Cao et al., 2022; Mao et al., 2020) or use the DCT positional embedding midway through the network, after using a GCN to encode the spatial dependencies between joints in a frame and a TCN (1D CNN across stacked graphs) to encode temporal dependencies (i.e. Li et al., 2021a). Mao et al. (2020) observe that human motion tends to repeat itself somewhat periodically (e.g. walking, talking, picking up and putting down, etc.) so they propose a motion attention mechanism which operates like a sliding window across the most recent frames to model the similarity between the current motion context and historical motion sub-sequences. Similar to searching a text for a key phrase, Mao et al. (2020) compare the last visible sub-sequence with a history of motion sub-sequences using attention on aggregated long-term temporal information from the history, before predicting future poses with a GCN. Similarly, Cao et al. (2022) improve this temporal attention module by implementing local attention (in addition to a CNN) separately to the 'last visible' (search key) sub-sequence and 'history of motion' (analogous to the full text) sub-sequence, and then put their outputs into a global attention module which measures the similarity between the two sub-sequences. The networks in both Cao et al. (2022), Mao et al. (2020) predict only a short time into the future, but this prediction can be concatenated with the input and run through the network again to iteratively produce further future timesteps.

Other methods which also perform open-loop prediction (i.e. iteratively producing further future timesteps by making small predictions, concatenating them with the input, and predicting again) include Jain et al. (2016), Wang et al. (2019), Gui et al. (2018). The advantage of breaking the long prediction sequence into smaller windows to predict a much shorter timestep into the future is that the difficulty of prediction is greatly reduced (Wang et al., 2019). Wang et al. (2019), for example, predict a small window of future poses at a time by training a reinforcement learning policy where the input state (defined as long sequences of previous pose observations and predictions) is encoded via a recurrent GRU network in order to predict output actions (short sequences of future pose vectors). Meanwhile, Jain et al. (2016) turn skeleton sequences into a spatiotemporal graph consisting of: three nodes for joints of each type of body part (spine, arm, and leg), four edges modeling the spatio-temporal interactions between them, and three more edges to learn the temporal patterns of each node. This maintains the spatial and temporal structure of the skeleton sequence while allowing joints with similar semantic functions to share weights (Jain et al., 2016). On the other hand, Gui et al. (2018) use a recurrent GRU network to iteratively encode the 3D positions and velocities of skeleton joints, as well as the class of the action being performed, in order to predict the future skeleton joints one frame at a time by taking previous outputs as part of the subsequent input. Incorporating velocities helps capture local motion information, while knowing the action class allows the model to roughly sketch out the course trajectory (Gopalakrishnan et al., 2019).

Similarly, Gopalakrishnan et al. (2019) incorporate velocity and action class through a bidirectional recurrent network where they encode the action label in the backwards direction, and input the velocities of each joint of the pose sequence into the forwards direction. Using the action class in the backwards direction to generate guide vectors for the forwards RNN forces the backwards part of the model to depend more heavily on information from the future. Another advantage of this structure is that it offers better regulation than using mechanisms like drop-out (Gopalakrishnan et al., 2019). However, unlike the methods described above, Gopalakrishnan et al. (2019) show that open-loop methods fail to make good predictions over long time horizons due to drift and accumulation of error. Therefore, their method performs both open loop prediction (iteratively predicting small timesteps) and closed loop prediction (directly predicting a large timestep into the future), and dynamically combines the losses from these two methods as described in the Training Methods section.

Modeling Methods. While some methods use GCNs (e.g., Mao et al., 2019, 2020; Li et al., 2021a; Cao et al., 2022), others rely on recurrent architectures (e.g., Jain et al., 2016; Gopalakrishnan et al., 2019; Gui et al., 2018), while Wang et al. (2019) employs inverse reinforcement learning.

GCNs offer a natural way to learn the dependencies between the different joint trajectories since they take the connected joint positions, connect them vertically across time to future corresponding joint positions, and learn the graph connectivity during training via convolutional networks (Mao et al., 2019). While Mao et al. (2019) forms the initial graph by replicating the last pose of the history as a stand-in for future poses, and runs the GCN on the spatiotemporal graph containing the combined history and future stand-ins, Mao et al. (2020) and Cao et al. (2022) predict only a short timestep into the future and use previously predicted poses to iteratively replace the 'stand-in' poses. Li et al. (2021a), however, use a TCN (defined in the Input Encoding section) on top of the output of the GCN, within the decoder, to capture long-term temporal dependencies. Furthermore, final TCN layer produces a series of future skeleton embeddings all at once, thereby eliminating the error accumulation problem (i.e. open-loop prediction issue mentioned in Input Encoding section) in a different way.

Of the recurrent architectures, Gopalakrishnan et al. (2019) uses a bidirectional GRU that runs both forwards and backwards in time in order to decompose the problem into a two-level process, as has been done in neural dialogue modeling (Gopalakrishnan et al., 2019). The backwards GRU encodes the action type into its synaptic weights to create a coarse trajectory for the forwards GRU, which refines that prediction based on either the input data or its own closed-loop predictions. Meanwhile, Jain et al. (2016) use an LSTM network on each node/edge of the graph to aggregate temporal information regarding each body part type. Gui et al. (2018), on the other hand, use a recurrent encoder-decoder GRU network to iteratively generate future poses, but they create a discriminator network to differentiate between the generated pose sequence and the ground truth pose sequence, and use this adversarial loss to train the network. One advantage of this method is that due to the discriminator, the distribution of generated pose sequences closely resembles that of the ground truth sequences, which reduces the error accumulation problem inherent in open-loop prediction methods while still allowing the recurrent generator to perform the relatively easier task of iterative open-loop prediction.

These advantages also apply to the method in Wang et al. (2019), which also uses a discriminator in its adversarial framework. This framework, however, (which is an adversarial generative imitation learning framework), uses a markov decision process (MDP) to learn the optimal policy that takes the state (the pose history sequence) as input and iteratively outputs an action (the future pose), while a discriminator attempts to differentiate between the generated pose sequence and the ground truth. The main advantage of using an MDP is that it generalizes well over unseen domains and maintains strong sequential correlation across different actions, which enforces the learning of long-term dependencies (Wang et al., 2019).

Training Methods. While most methods simply take the difference between the ground truth and predicted joint locations (in angular coordinates) to be the loss, (e.g., Mao et al., 2019, 2020; Cao et al., 2022; Jain et al., 2016), Jain et al. (2016) also adds noise to the input frames during training to keep the "forecasted motion close to the manifold of human motion" (Jain et al., 2016). Gopalakrishnan et al. (2019) use a similar squared difference loss but in order to train the model to perform closed-loop prediction on short-term as well as long-term timescales, they run the model twice in each iteration (once in open-loop mode and once in closed-loop mode) and dynamically weight the losses from the open-loop and closed-loop predictions such that more importance is placed on the open-loop loss at the beginning of the training, and on the closed-loop loss at the end. This dynamic loss is shown to perform better on long-term human motion prediction than other long-term-focused losses such as noise scheduling, auto-conditioning, and sampling loss (Gopalakrishnan et al., 2019). Li et al. (2021a) also uses the angular error, but in quaternion coordinates, and additionally try to predict the action class label in order to include the classification loss into the loss function. One MLP action recognition classifier tries to predict the action class from the input encoding, while another branch applies the input embedding architecture to the predicted future pose sequence and uses a second action recognition classifier on that embedding (Li et al., 2021a). The ablation study done by Li et al. (2021a), which shows improved performance through the addition of the classification loss, confirms the intuition that high-level human action categories typically guide low-level joint motions.

Finally, both Wang et al. (2019) and Gui et al. (2018) use adversarial loss from the discriminator to train the generation of future pose sequences. Wang et al. (2019) also pre-trains its policy generator network with an initial minimum distance error loss between the ground truth and predicted pose sequences to "first recover an estimate of the reward signals underlying the MDP from the expert's demonstration," and then uses the GAIL framework to effectively "optimize the agent's policy using the rewards signals that it recovered" (Wang et al., 2019).

Discussion As shown in Table 5, the highest performing methods seem to be the ones which include a DCT within the input encoding method, and a GCN within the modeling method. This may be due to the fact that the DCT effectively encodes low-frequency temporal information, and the fact that GCNs effectively propagate spatiotemporal dependencies into the future, but more work needs to be done to confirm whether one or both of these methods are leading to the high performance. Another notable distinction of the GCN methods (i.e. Mao et al., 2019, 2020; Cao et al., 2022) is that they all use L_1 joint angle error loss, which is correlated with the MAnE metric, but as discussed above, every method uses some form of joint angle error loss except for Gui et al. (2018). The reason other training methods in Table 5 don't state this explicitly is due to lack of space, since the table prioritizes the most unique aspects of each model.

Another interesting observation is that even though Gui et al. (2018) and Gopalakrishnan et al. (2019) use action class as an input while Li et al. (2021a) use it in the loss (making action class knowledge unnecessary during inference), Li et al. (2021a) still obtain higher performance, which further demonstrates the robustness of this method. One caveat to note, however, is that the metrics shown in Table 5 are for various timeframes within the *walking* class, a relatively periodic action class, and that performance comparisons on less periodic action classes may differ.

Model	History Encoding Method	Map Encoding Method	Modeling Method	Training Method	Social Interaction Modeling	Datasets				
						ETH	HOTEL	UNIV	ZARA1	ZARA2
Mangalam et al. (2020a)	trajectory heatmap	segmentation map	U-net	preliminary pred loss	-	0.28/ **0.33**	0.10/ 0.14	0.24/ 0.41	0.17/ 0.27	0.13/ 0.22
Wong et al. (2021)	DFT	MLP	Transformer +GCN	preliminary pred loss + KL loss	MLP	0.22/ 0.35	0.10/ 0.15	0.26/ 0.44	0.17/ 0.29	0.15/ 0.24
Zhou et al. (2021)	MLP+ CVAE	-	recurrent CVAE	preliminary KL loss	masked Transformer	**0.19/** 0.35	**0.06/ 0.07**	**0.13/ 0.21**	**0.06/ 0.07**	**0.05/ 0.08**
Wang et al. (2022)	GRU	-	CVAE + GRU	preliminary pred loss + KL loss +variety loss	-	0.35/ 0.65	0.12/ 0.24	0.20/ 0.42	0.12/ 0.24	0.10/ 0.21
Salzmann et al. (2021)	LSTM	CNN	CVAE + GRU	KL loss	graph LSTM+ attention	0.39/ 0.83	0.12/ 0.21	0.20/ 0.44	0.15/ 0.33	0.11/ 0.25
Huang et al. (2019)	velocity LSTM	-	LSTM	variety loss	GAT+ LSTM	0.65/ 1.12	0.35/ 0.66	0.52/ 1.10	0.34/ 0.69	0.29/ 0.60
Kosaraju et al. (2019)	MLP+ LSTM	CNN+ attention	LSTM GAN	adversarial + KL loss + noise loss	GAT	0.69/ 1.29	0.49/ 1.01	0.55/ 1.32	0.30/ 0.62	0.36/ 0.75
Sadeghian et al. (2018b)	LSTM	CNN+ attention	LSTM GAN	adversarial	proximity attention	0.70/ 1.43	0.76/ 1.67	0.54/ 1.24	0.30/ 0.63	0.38/ 0.78
Zhao et al. (2019)	LSTM	CNN	LSTM GAN	adversarial	U-net	1.01/ 1.75	0.43/ 0.80	0.44/ 0.91	0.26/ 0.45	0.26/ 0.57
Alahi et al. (2016)	Social LSTM	-	Social LSTM	-	social pooling	0.50/ 1.07	0.11/ 0.23	0.22/ 0.48	0.25/ 0.50	0.27/ 0.77
Hasan et al. (2018)	trajectory kinematics	-	Energy minimization	Nelder Mead simplex	collision avoidance term	-	-	-	0.30/ 0.59	0.26/ 0.60

Table 6: **Motion Prediction.** Summary of **Trajectory Prediction** methodology and performance on the ETH (Pellegrini et al., 2010) (ETH, HOTEL, UNIV) and UCY (ZARA1, ZARA2) (Lerner et al., 2007) datasets, in average distance error (ADE) and final distance error (FDE) of the best of 20 predicted samples, formatted ADE/FDE. Dashes in the performance indicate that this metric was not included in the paper. Dashes in Map Encoding indicate that the map is not used as an input, and in Interaction Modeling, that social interaction is not explicitly modeled. Abbreviations are: DFT, discrete fourier transform; MLP, multi-layer perceptron; CVAE, conditional variational auto-encoder; KL, Kullback-Leibler; GAT, graph attention network; Transformer, attention-based network from Vaswani et al. (2017). Bolded numbers indicate best performance and underlined numbers indicate second-best performance.

2.3.2 Trajectory Prediction

Trajectory prediction is the task of predicting the future locations of dynamic agents, either as regressed image bounding boxes (e.g., Yao et al., 2019), top down coordinates in meters (e.g., Sadeghian et al., 2018b), or top-down coordinates on a rasterized map in pixel coordinates (e.g., Dwivedi et al., 2020). Meanwhile, agent history is typically represented as a set of past location coordinates (e.g., Sadeghian et al., 2018b) and/or ego-centric images (e.g., Chandra et al., 2020), but many other modalities can also be added to the input, like semantic or image based maps (e.g., Salzmann et al., 2021), and visual features (e.g., Haddad & Lam, 2021) taken from maps or ego-centric images. The top-down perspective Trajectory Prediction example in Figure 4 shows the map in grey with various map features such as buildings or vegetation plotted on the map, while the agents' histories and futures are shown as red/yellow arrows in the left and right hand side images, respectively.

In this section, we will briefly review the prominent trajectory prediction papers listed in Table 6 and compare their performance on the the most commonly benchmarked trajectory prediction dataset using the most popularly used trajectory prediction metrics. The methods listed in Table 6 were selected to maximize the variety of techniques presented, show the highest performing models (which is Zhou et al., 2021), and recognize methods that pushed the state-of-the-art by a large margin (such as Sadeghian et al., 2018b; Salzmann et al., 2021). The methods surveyed typically use two types of inputs (history and map), from which social interactions between agents can be deduced, and then combined along with the history and map encodings into a compact representation, before being decoded into predicted future trajectories by the main part of the modeling method. Therefore, in this section, we separate the methodologies of the works surveyed into: the history encoding method, map encoding method, social interaction modeling method, modeling method, and training method.

Datasets. The ETH-UCY dataset (Pellegrini et al., 2010; Lerner et al., 2007) is the most commonly benchmarked trajectory prediction dataset. It contains only pedestrians, provides 3.2 seconds of history over 8 frames and 4.8 seconds of future over 12 frames. The most popular trajectory prediction metric is the 'Best of 20' average distance error (ADE), and final distance error (FDE). As is described in Kothari et al. (2021), the average distance error is the mean of the differences between the predicted path and the ground truth path in each of the future frames (in this case, the mean across the 12 future frames), whereas the final distance error is the difference between the predicted and ground truth paths only in the frame that is furthest into the future (in this case, the 12th and last future frame). One issue with defining a single ground truth path per trajectory, however, is that one history sequence could yield multiple plausible paths. For example, when encountering an obstacle directly in their path, an agent could go either left or right when walking around it. Therefore, many methods evaluate their models using the multimodal evaluation metrics 'Best of N' or 'top-N', where the N most likely paths are predicted, and the path which is closest to the ground truth is taken to be the predicted path from which ADE and FDE metrics are measured (Kothari et al., 2021). While different methods can use different values of N, most of the methods that use the ETH-UCY dataset, to our knowledge, use N=20 and quote their metrics as 'Best of 20 ADE/FDE'. Notably, however, the TrajNet++ Trajectory Forecasting Challenge (Kothari et al., 2021) recommends an N value of 3, and makes a strong case for why 20 is too many, since ”a model outputting uniformly-spaced predictions, irrespective of the input observation, can result in a much lower Top-20 ADE/FDE.” However, in order to better compare existing methods, we share the ‘Best of 20’ ADE/FDE metrics quoted in the works which are discussed.

History Encoding Methods. History is typically encoded via an LSTM network (e.g., Huang et al., 2019; Alahi et al., 2016; Sadeghian et al., 2018b; Zhao et al., 2019; Kosaraju et al., 2019; Salzmann et al., 2021) since LSTMs have been shown to successfully learn and generalize the properties of isolated sequences like handwriting and speech (Alahi et al., 2016). LSTMs typically allow one position at a time to be processed by the LSTM, and output a final embedding which encodes entire trajectory. While some methods use a single LSTM with shared weights to encode the positions within the trajectory of each agent in the scene separately (e.g., Sadeghian et al., 2018b; Zhao et al., 2019; Salzmann et al., 2021), Huang et al. (2019) separately encode the velocity instead of position of each agent and Alahi et al. (2016) use a ‘Social LSTM’, which also uses a separate LSTM network for each agent’s trajectory, but allows the LSTMs of subsequent timesteps to also be influenced by the hidden layers of multiple other agents (which are in close proximity) from previous timesteps. This method uses the Social LSTM to model the tendency of pedestrians to adapt their motion based on the behaviour of other people in their vicinity Alahi et al. (2016), and is described in more detail within the Social Interaction Modeling section. Kosaraju et al. (2019) also encode each pedestrian’s trajectory via an LSTM, but they first embed each position within a trajectory into a higher dimension using an MLP before inputing it into the LSTM.

Meanwhile, Wang et al. (2022) and Zhou et al. (2021) use other recurrent architectures to encode the trajectory. Zhou et al. (2021) also first use an MLP, but the apply two different MLPs: one to encode the entire history trajectory, and one to encode the next future ground-truth position that’s one timestep into the future. Then, these are input into a recurrent CVAE which learns a latent variable representation that helps predict the relationship between the history and its next future position. The advantage of using this CVAE-based embedding is that CVAEs have been shown to be very good at problems which suffer from multimodality; they can output multiple divergent (not precise), but accurate and likely predictions from a single input. On the other hand, Wang et al. (2022) uses a simple GRU architecture which not only takes in the inputs (i.e. the agent’s position, velocity, and acceleration at each timestep in history) and hidden states of past timesteps, but also takes into account the preliminary future predictions made in past timesteps. This architecture is based on the idea that people regularly adjust and optimize their intended paths, and that previous intentions can impact one’s perception of the present and the development of new future plans (Wang et al., 2022). The benefit of taking past preliminary predictions into account in future input embeddings is further validated through an ablation study which shows the improvement in metrics when this connection between past predictions and future inputs is added.

Finally, some methods use non-recurrent methods to encode agents’ historical trajectories (e.g., Hasan et al., 2018; Mangalam et al., 2020a; Wong et al., 2021). While Hasan et al. (2018) incorporates the position and gaze direction of all agents in the scene, at each timestep, into a simple kinematic model that calculates their velocity, acceleration, and orientation at each point in time, Mangalam et al. (2020a) plots each history position onto a birds-eye-view heatmap (which are the same dimensions as the input map) where the values decrease inverse proportionally to the distance from that position, and then concatenates these heatmaps with the input map encoding to put through a U-net (Ronneberger et al., 2015) style architecture. This non-quantized way of representing past histories allows aleatoric uncertainty (noise) to be incorporated within the system, and facilitates the combining of history and map encodings when predicting the future (Mangalam et al., 2020a). Wong et al. (2021), however, encodes history coordinates by their discrete fourier transform (DFT) frequencies (similar to the DCT in human motion prediction) by applying a 1D-DFT on each dimension of the trajectory (x and y), obtaining a magnitude and phase sequence for each dimension, and embedding the concatenated magnitude and phase sequences into a higher dimension through an MLP. Since pedestrians typically make a coarse overall motion decision first, and then respond to potential emergencies (like interactive behaviors) with quicker maneuvers, Wong et al. (2021) use the DFT frequencies to take advantage of the fact that low-frequency portions in the spectrums of agents' observed trajectories reflect coarse motion trends and can potentially predict global future motion, while the high-frequency portions reflect quicker movements, by separating the motion forecasting into a coarse trajectory prediction which is then refined by a fine interpolation network (Wong et al., 2021).

Map Encoding Methods. Most map encoding methods use a CNN (e.g., Sadeghian et al., 2018b; Zhao et al., 2019; Kosaraju et al., 2019; Salzmann et al., 2021). Kosaraju et al. (2019) and Sadeghian et al. (2018b) use a VGG network to distill the top-down image map into a single vector (where Sadeghian et al., 2018b pre-train it on the Image-Net dataset from Russakovsky et al., 2015 and fine-tune it on the task of scene segmentation) in order to apply attention over the joint map and history encodings, while Zhao et al. (2019) use a CNN autoencoder in order to keep the height and width dimensions the same and Salzmann et al. (2021) create their own CNN architecture and then put the output through a fully connected layer. Similarly, Mangalam et al. (2020a) semantically segment the map into classes (determined according to the affordance provided by the surface to an agent for actions such as walking, standing, running etc.) using a U-net architecture, and Wong et al. (2021) use an MLP on the map image to highlight where on the image social interactions are most likely to take place. Finally, found that including map information (by rasterizing the agents' trajectories, overlaying it onto the map, and then applying a CNN to extract features to input into the initial decoder's hidden state) significantly improved results on a different dataset (NuScenes, Caesar et al., 2020), but this experiment wasn't done on the ETH/UCY datasets.

Social Interaction Modeling. While social interaction within the other sub-tasks presented thus far has been studied to a certain extent, as shown in Table 9, the most of the methods whose performances were compared in previous sections did not explicitly model social interaction (with the exception of Yao et al. (2021) and Wu et al. (2021), whose relational scene graphs capture human-object interaction as well as human-human social interaction). However, many of the trajectory prediction methods add specific components within the neural network in order to capture social interaction. Therefore, we present a review of those methods here.

Initially, social pooling modules proposed by Alahi et al. (2016) performed social interaction modeling by allowing each LSTM cell to receive pooled hidden-state information from the LSTM cells of neighbors (within a certain spatial distance), where the pooling step performed grid based pooling by summing the hidden states of the neighbors within each grid location. This allows models to aggregate information from multiple neighbors, while simultaneously dealing with the fact that every person has a different number of neighbors and that in very dense crowds, this number could be prohibitively high (Alahi et al., 2016). Zhao et al. (2019) effectively performs spatial pooling across neighbors, but in a different way: in this model, the map encoding and past trajectories' encodings are spatially concatenated such that the history encodings are stacked on top of the spatial location on the map in which they currently reside, like stacks of pepperoni on a pizza (if multiple agents are placed into the same cell in the tensor due to discretization, element-wise max pooling is performed, while cells which contain no agents are initialized to 0). Then, a U-net (Ronneberger et al., 2015) type architecture is applied at different spatial scales in order to learn to represent interactions among multiple agents and between agents and the scene context, while retaining spatial locality, and the transformed stacks of history encodings (which now contain the interaction, history, and constraint features for the corresponding agent), are sliced out (Zhao et al., 2019).

Social pooling methods have dominated social interaction modeling within trajectory prediction for many years, but social pooling models have some disadvantages: they discard uniqueness when performing pooling, and impose human biases like measuring the impact of neighbors based on their euclidean distance (Kosaraju et al., 2019). Furthermore, many works have shown that when considering the influences of other pedestrians, each pedestrian in the scene is necessary (Huang et al., 2019). Therefore, attention modules like graph attention networks (GATs) have been used to deal with these issues. For example, Huang et al. (2019) uses one GAT for each frame of the history by forming a graph between each pedestrian and every other pedestrian present at that timestep. Huang et al. (2019) use the pedestrians' hidden layer representations at each timestep as the nodes, and the existence of interactions between them to represent the edges. Then, the features of each graph node are computed by attending over its neighbors (Huang et al., 2019). Finally, these frame-by-frame social interaction graphs are combined temporally by using another set of LSTMs which operate on one graph node (i.e. one pedestrian) at a time, and output a socially aware embedding for each pedestrian (Huang et al., 2019). Kosaraju et al. (2019) also use GATs, but they use the whole history encoding of each pedestrian to make a fully connected graph, and apply stacked graph attention layers to produce an embedding of each pedestrian as an attention-weighted sum of the neighbors they interact with. Salzmann et al. (2021), however, use a graph LSTM and then apply an attention layer on top of it. Similar to Kosaraju et al. (2019), each node represents the encoded history of each pedestrian, while edges are aggregated from all other agents using an element-wise sum (in order to handle a variable number of neighbors while preserving count information) and then fed into an LSTM whose weights are shared across edges (Salzmann et al., 2021). Finally, all edges that connect to a node are combined using an additive attention model to get the social influence vector for that node (Salzmann et al., 2021). These graph-based attention networks are advantageous because they allow information to be aggregated from each neighbor by assigning different importance to different nodes, without imposing a biased structure on the data (Huang et al., 2019).

Zhou et al. (2021) uses a graph-based component in their Transformer-based social interaction model in order to mask out the influence of neighbors that are too far away, or don't overlap in time, but the main part of their social interaction model applies a Transformer network to the combined history and predicted futures of all the agents in the scene (by learning the query/key vectors, applying the mask to their output, and then attending that over the value vector) to learn the relationships between the trajectories of different agents, and then refines the previously predicted futures accordingly. Finally, Sadeghian et al. (2018b) merely use attention with no graphs. They first create a joint feature vector from the input encodings of all the other agents in the scene, ranked by their distance from the agent whose future is being predicted (since sorting can keep the uniqueness of the neighbors unlike the average or max functions used in social pooling methods). Then, this joint feature vector and the hidden state of the decoder LSTM of that agent are input into a soft attention module, combined with the map encoding, and used as an input to the subsequent timestep's decoder LSTM. Since humans pay more attention to close obstacles, upcoming turns and people walking towards them, than to the buildings or people behind them, these attention modules aim to let the model to focus more on the salient regions of the scene and the more relevant agents when predicting the future state of each agent (Sadeghian et al., 2018b).

Finally, Wong et al. (2021) use a multi-layer perceptron (MLP) to combine the scene map and neighbor trajectories into a heatmap (with the same height and width as the input map) that highlights where on the map interactions are more likely to occur. Meanwhile, Hasan et al. (2018) use a simple collision avoidance term in their energy minimization method to minimize the likelihood of predicting futures where pedestrians collide.

It should be noted that while this survey only records the performance of these methods on the ETH-UCY dataset, many of these papers (e.g., Zhao et al., 2019; Salzmann et al., 2021; Wang et al., 2022) also evaluate their methods on vehicle datasets such as nuScenes (Caesar et al., 2020) and show similar trends in methodology, where social interaction models that use attention and GATs outperform social pooling methods.

Modeling Methods. Most of the methods shown in Table 6 use recurrent methods to predict the future timesteps iteratively, by re-incorporating past predictions to obtain subsequent predictions (e.g., Huang et al., 2019; Zhao et al., 2019; Kosaraju et al., 2019; Sadeghian et al., 2018b; Alahi et al., 2016; Salzmann et al., 2021), while some methods predict all the future timesteps at once (e.g., Mangalam et al., 2020a; Hasan et al., 2018; Wong et al., 2021). Others use a combination of both: Wang et al. (2022) repeatedly make preliminary predictions of all future timesteps at once (at every future timestep), and use each set of preliminary full predictions to make the next timestep's singular prediction, while Zhou et al. (2021) make initial step-by-step predictions, one timestep at a time, and then refine the resulting full predictions all at once through their 'socially aware regression module.'

Recurrent modeling methods typically use LSTM based decoders to predict future trajectories (e.g., Huang et al., 2019; Zhao et al., 2019; Kosaraju et al., 2019; Sadeghian et al., 2018b; Alahi et al., 2016) while a few use gated recurrent units, GRUs (e.g., Wang et al., 2022; Salzmann et al., 2021). Alahi et al. (2016) uses the same Social LSTM input encoding architecture (which uses pooled information from other neighboring agents as an additional input into the subsequent timestep's LSTM), as a decoder to predict the next position of an agent (from either the last ground truth positions, or the last previously predicted positions of the agents in the scene) in order to incorporate social information into future predictions. Meanwhile, Huang et al. (2019) concatenate the socially aware embedding of each agent's history (derived by applying a temporal LSTM to corresponding nodes of the frame-by-frame GATs) with the non-socially aware input encoding of each agent's history (derived from a simple LSTM) in order to model the temporal correlations of interactions, and the motion pattern of each pedestrian, respectively, and combine them. Finally, Huang et al. (2019) use a decoder LSTM network to iteratively predict the future trajectory. Using a decoder LSTM with shared weights forces the network to generalize well, even when factors like the number of agents in a scene, vary (Zhao et al., 2019).

For similar reasons, Zhao et al. (2019) also add the socially (and spatially) aware embedding of each agent's history (derived from slicing each agent's history stack out of the U-net's output tensor), to the initial LSTM-based input encoding and also use an LSTM decoder to iteratively predict future positions, but they additionally employ a discriminator (which uses the same input encoding and U-net architecture, followed by FCNs) to differentiate between real and generated trajectories.

GANs and discriminators are also used by Kosaraju et al. (2019), Sadeghian et al. (2018b). Sadeghian et al. (2018b) concatenates the map encoding (which has been attended over by the history encodings), the social interaction encoding (output by the soft attention social interaction module), and a random gaussian noise vector (which can be fixed and/or altered in order to generate multimodal outputs), for each agent, to input into an LSTM which iteratively generates their subsequent future positions. Meanwhile, the discriminator, a simple LSTM, tries to distinguish between synthetic and real trajectories. Kosaraju et al. (2019) concatenates the nodes of all agents in the social GAT, and adds it to each agent's history embedding, as well as the map embedding and a random gaussian noise vector (for generating multimodal outputs), and puts the combined output through a decoder LSTM to output that agent's future. Kosaraju et al. (2019) then uses two discriminators, one which uses an LSTM to discriminate between real and fake trajectories, and one which uses the combined history, social, and map embeddings of the trajectories to differentiate. The purpose of using two different discriminators in Kosaraju et al. (2019) is to encourage the production of realistic local and global trajectories, which both adhere to the manifold of existing trajectories and comply with the scene information given.

Combining history, map, and social interaction encodings to input into the decoder, as is done in the previous two methods, is helpful because it 1) suppresses the redundancies of the input data, allowing the decoder to focus on the important features, and 2) deals with the complexity of modeling all agents while adding interpretability to predictions (Sadeghian et al., 2018b). Although Kosaraju et al. (2019) concatenate the map, social interaction, and history encoding, Sadeghian et al. (2018b) only concatenate their map and social interaction encoding because Sadeghian et al. (2018b) consider history information to be already contained within the social interaction encoding. Finally, the extra gaussian noise factor used by both methods adds the following advantage typically enjoyed by conditional GANs: the predicted trajectory becomes conditioned on the noise vectors such that sampling n random noise vectors will produce n possible future trajectories. While Sadeghian et al. (2018b) sample the noise vector from a multivariate normal distribution conditioned on the social interaction and map encodings, Kosaraju et al. (2019) sample from a gaussian distribution whose mean and variance are dictated by an MLP network which takes in the history encodings. The purpose of using this random noise vector is to encourage generalization towards a multimodal distribution despite only having access to single samples from single modes of behavior, and this is done by constructing a reversible mapping between outputted trajectories and latent noise vectors that represent pedestrian behavior in a scene (Kosaraju et al., 2019).

The other major method used within the recurrent modeling techniques is the GRU. For similar reasons to the above methods, Salzmann et al. (2021) concatenates the map encoding, history encoding, and social influence vector into a single learned feature representation, and then uses this representation as the input to a CVAE model in order to learn a latent space embedding, which is then used to predict future positions iteratively using a GRU. Just like the random noise vector described above and used in Kosaraju et al. (2019), Sadeghian et al. (2018b), the goal of the CVAE is to explicitly handle multimodality and allow the latent space embedding to learn high level latent behavior (Salzmann et al., 2021). However, unlike the random noise vector described above, the CVAE accomplishes this by using the ground truth future trajectory directly within the model to learn the latent space embedding. While one branch of the model estimates the latent space embedding using the concatenated feature representation, a second branch estimates the latent space embedding using the ground truth future trajectory, as well as the feature representation, and the CVAE loss (KL loss, further described in Training Methods) minimizes the difference between the two latent space embedding estimates. During inference, however, only the branch which uses just the feature representation is employed to create the latent space embedding used by the GRU. Wang et al. (2022) also uses a combination of CVAE and GRU, but here, the input to the CVAE consists only of the history embedding (and, during training, the ground truth future trajectory), since this method uses no map or social interaction model whatsoever. Furthermore, at every timestep in the future prediction, Wang et al. (2022) produce preliminary guesses at what the positions of subsequent future timesteps will be (e.g. at timestep $t + i$, which is i timesteps past the last history timestep, they produce preliminary guesses of the positions from timestep $t + i + 1$ until timestep $t + f$, where f is the maximum number of future timesteps). Then, these guesses (compressed into a single representation via attention), as well as the latent space embedding produced by the CVAE, are used as input to decoder GRU cells, which make one final prediction for the timestep $t + i + 1$. The intermediate predictions help guide the final prediction by modeling the way people regularly adjust and optimize their intentions based on ever-changing developments of new future plans, while the attention layer over these intermediate goals learns the different impact each goal has on the prediction (Wang et al., 2022). The use of GRUs compared to MLPs and CNNs was shown to be optimal in an ablation study by Wang et al. (2022), and as before, the CVAE learns latent information to produce realistic multimodal outputs. To encode the ground truth future trajectories that are used within the CVAE, Salzmann et al. (2021) employs a bi-directional LSTM while Wang et al. (2022) uses a fully-connected layer.

Meanwhile Zhou et al. (2021) uses only a CVAE, arranged in a recurrent scheme: at each timestep $t + i$, which is i timesteps past the last history timestep, the history from timestep i (i timesteps past the first history timestep) until timestep $t + i$ is encoded into the history embedding by an MLP, while the future position at timestep $t + i + 1$ is encoded into the ground truth embedding by another MLP, and a CVAE is used to learn the latent relationship between the history embedding and ground truth embedding for that timestep. Then, the learned latent relationship is used to iteratively predict preliminary guesses for the future of each agent in the scene using only the history (Zhou et al., 2021). Finally, the social interaction module uses the histories and preliminary guesses of each agent to predict offsets that are added to the preliminary guess of each agent to produce the refined final trajectory predictions, in order to prevent impracticalities such as colliding trajectories (Zhou et al., 2021). This two-stage predictor mimics the structure proposed in Wang et al. (2022) and for similar reasons, aids prediction by providing a general goal trajectory which is later adjusted by environmental (i.e. social) constraints. The purpose of using recurrent CVAEs is to reduce the accumulation of error (i.e. drift) inherent in open-loop prediction models (further described in the input encoding methods section within 2.3) which use the outputs of previously predicted timesteps to predict subsequent timesteps (Zhou et al., 2021). One reason for this drift is due to the multimodality of prediction (e.g. if a plausible but incorrect mode of the future is predicted at a previous timestep, and used to predict the next timestep, other plausible and possibly correct modes are automatically ruled out, causing error which can accumulate when this process is repeated). Since CVAEs help to handle multimodality by using a distribution of plausible predicted futures at every timestep, recurrently predicting subequent timesteps using CVAEs can significantly reduce accumulated error (Zhou et al., 2021). This is shown to be the case through an ablation study that shows how the average displacement error increases significantly at further timesteps into the future (i.e. longer term forecasting) for PECNet (previous state-of-the-art method), while the average displacement error stayed mostly the same at all future timesteps for Zhou et al. (2021).

However, there are other methods which don't use recurrent processes at all. Mangalam et al. (2020a), for example, uses a U-net architecture where the concatenated semantically segmented map and history trajectory heatmaps get downsized into a compact embedding through the encoder, and then upsized into predicted future trajectory heatmaps through the decoder to produce one heatmap for the predicted position at each future timestep. However, these predicted future positions are merely used as preliminary predictions (similar to Wang et al., 2022), and are input back into a secondary decoder which upsizes the compact embedding (and preliminary predictions) into the final predicted future trajectory heatmaps (which is also one heatmap per future timestep). In order to insert the preliminary predictions tensor into the secondary decoder at every stage of the upsampling, the preliminary predictions are downsampled to match the tensor sizes at various levels of the network, and concatenated with those tensors at each stage (Mangalam et al., 2020a). However, one issue with this method is that the modes of the future trajectory heatmaps may not be continuous, as they may be predicting a specific timestep from different multimodal futures. Therefore, during inference, the prediction for the last future timestep (i.e. the 'goal') is first set by using the softargmax function (Mangalam et al., 2020a) across the predicted heatmap of the last timestep, and then the second to last timestep's prediction is made by 1) forming a gaussian heatmap around where the agent would be at the second to last timestep if it were to follow a straight line from its current position (last history timestep) to its 'goal', 2) multiplying the gaussian heatmap with the predicted heatmap for the second to last timestep, and 3) taking the resulting argsoftmax of this distribution to be the prediction of the second to last timestep. This process is then performed iteratively, backwards in time (using the previously predicted timestep as the 'goal') to get the remaining predictions until the first future timestep (Mangalam et al., 2020a). To produce multiple (i.e. N) possible future trajectories, 1000 initial 'goal' positions are sampled from the last future timestep's heatmap, and K-means is used to cluster those into N means. The goal of this method is to model two different types of future undertainty: epistemic uncertainty caused by latent decision variables like long term goals and aleatoric variability from random unforeseeable variables such as environmental factors or handedness (Mangalam et al., 2020a). Similarly to Wang et al. (2022), the epistemic uncertainty is modeled by predicting the longest term futures (i.e. 'goals') first, while the aleatoric factor is modeled as the resulting distribution over the intermediate trajectory points for each goal (Mangalam et al., 2020a).

Another non-recurrent method, Wong et al. (2021), makes a similar attempt to perform two stage prediction: preliminary predictions are made by 1) concatenating the history embedding and map/interaction embedding, 2) applying a Transformer encoder/decoder and taking the penultimate layer's features as 'behavior features' 3) using a graph convolution layer to aggregate behavior features at different frequency nodes (in order to learn multiple activity features that represent agents' differing decision styles), and 4) applying an MLP to predict the magnitudes and phases in each dimension at several 'key' future timesteps. This results in a 'coarse' prediction which focuses on forecasting trajectories with low spatiotemporal resolution (Wong et al., 2021). To make the 'fine' prediction which reconstructs trajectories from keypoints sequences with a higher spatiotemporal resolution (on the frequency spectrum), they 1) concatenate the preliminary 'coarse' magnitude and phase predictions at key future timesteps with the map/interaction embedding, and 2) apply a Transformer encoder/decoder to produce the magnitudes and phases a each future timestep (Wong et al., 2021). Finally, trajectory positions can be obtained via an inverse DFT. Similar to Mangalam et al. (2020a), Zhou et al. (2021), Wang et al. (2022), this network attempts to make preliminary coarse predictions that guide the general direction and intention, before refining that trajectory to output a final prediction which takes into account environmental factors that may necessitate quick thinking and sudden changes to trajectory. In addition, this method goes one step further and uses the frequency spectrum of trajectories to help model the global motion trend, (which is encapsulated by low frequencies), as well as the environmentally or socially caused changes to this trend, which can be encapsulated by high frequencies, and predicted from the map/interaction embedding (Wong et al., 2021).

Finally, Hasan et al. (2018), a non-deep learning method, uses an optimization technique called energy minimization to incorporates trajectory kinematic information and gaze direction into an optimization cost function, which additionally adds a collision term to avoid predicting trajectories which collide into each other.

Training Methods. To solve their optimization problem, Hasan et al. (2018) use the Nelder-Mead simplex, a direct search method. Meanwhile, Huang et al. (2019), Wang et al. (2022) use a variety loss to model multimodality by increasing the diversity of trajectories predicted. The variety loss, also known as the 'Best-of-Many' loss, produces multiple possible trajectories by randomly sampling the prediction space and chooses the trajectory that has the smallest distance to ground-truth as the model output to compute the loss (Huang et al., 2019). Zhao et al. (2019), Sadeghian et al. (2018b), Kosaraju et al. (2019), on the other hand, use an adversarial loss based on the discriminator (on top of the typical L2 loss on generated trajectories). In addition, Kosaraju et al. (2019) adds a loss term which forces a one-to-one mapping between the latent noise vector (which is used to generate multimodal outputs) and the predicted trajectories by minimizing the difference between the latent noise vector and the latent encoder applied to the predicted output. This allows specific styles of multimodal outputs to be generated based on manipulation of the latent noise vector, and can be used to better visualize the predicted distribution.

The most common extra loss term used, however, is the Kullback-Leibler divergence loss (KL loss) term, which forces the distributions of two latent feature vector to be the same (e.g. Salzmann et al., 2021; Wong et al., 2021; Zhou et al., 2021; Kosaraju et al., 2019; Wang et al., 2022). This loss term can be used in two different ways: Salzmann et al. (2021), Zhou et al. (2021), Wang et al. (2022) use it to train their CVAEs while Kosaraju et al. (2019), Wong et al. (2021) use it as a minor loss term to force parts of their network to resemble a univariate gaussian. While Kosaraju et al. (2019) use KL loss to make sure the generated latent noise vector resembles noise drawn from a random Gaussian, and Wong et al. (2021) minimize the KL divergence between the activity features (output nodes of the GCN) and a gaussian with a mean of 0 and variance of 1, KL loss is typically used to train CVAEs to make sure that the latent feature embedding which is encoded from just the historical input has the same distribution as the latent feature embedding encoded using both the historical input and the ground truth future. This is why CVAEs use the ground truth future positions within their network and not just during back-propagation. Although there are two different networks being trained (one which uses only historical input and one which uses the ground-truth future), only one (the historical input network) is used during inference, which is why the networks need to be trained such that they produce the same latent representation as each other no matter what the input is.

Finally, the networks which perform two-stage prediction (e.g. Mangalam et al., 2020a; Wong et al., 2021; Wang et al., 2022; Zhou et al., 2021) also employ a loss on their preliminary predictions in order to train their preliminary prediction network. While Mangalam et al. (2020a) represents the ground truth future as a Gaussian heatmap centered at the observed points with a predetermined variance, and uses a binary cross entropy loss between the ground truth heatmaps and the preliminary predicted heatmaps (as well as another loss between the ground truth heatmaps and the final predicted heatmaps), Wang et al. (2022) utilizes two root-mean-squared-error (RMSE) losses between the 1) preliminarily predicted points and ground truth points, and 2) between the final predictions and ground truth points. Similarly, Wong et al. (2021) uses two L2 losses: one between the predicted 'coarse' keypoints (which are calculated using an inverse DFT on the predicted 'coarse' magnitudes and phases) and the corresponding ground truth values, and one between the final 'fine' trajectory and its corresponding ground truth. Zhou et al. (2021), however use the iterative KL divergence loss of their preliminary predictions as their preliminary prediction loss, and then apply an L2 loss between their final predictions and the ground truth, as well as an additional L1 loss between the predicted offsets and the actual offsets (predicted by their social interaction based trajectory refinement module).

Discussion. As can be seen in Table 6, all of the methods which have the best or second-best performance on the metrics shown use either a CVAE-based network (e.g.

Salzmann et al., 2021), or a two-stage prediction network (which makes preliminary predictions and then refines them into final predictions) (e.g. Mangalam et al., 2020a; Wong et al., 2021), where Wang et al. (2022) and Zhou et al. (2021), the two best performing methods, employ both CVAEs and a two-stage prediction network. Furthermore, neither of these methods use any map information for the ETH/UCY datasets, and Wang et al. (2022) additionally doesn't even include a social interaction modeling method. In fact, many of the best performing methods do not use a map (e.g. Wang et al., 2022; Zhou et al., 2021) and/or have no explicit social interaction modeling method (e.g. Mangalam et al., 2020a; Wang et al., 2022), although Wang et al. (2022) nevertheless show that applying a map encoding method from another paper significantly improves their results on the NuScenes dataset. Finally, most of the best methods also employ KL divergence losses.

These observations lead to the following conclusions. First of all, it seems that applying KL divergence loss, especially in combination with CVAE modeling methods, lead to strong performance. Since this task uses a best-of-20 evaluation metric (by measuring the error of the best trajectory, out of 20 predicted trajectories, that has the smallest distance to ground-truth), it follows that successful modeling of multimodal prediction (which the CVAE and KL loss specialize in), should lead to better performance. Another advantage of both the CVAE method, and the two-stage prediction method, is that they both reduce the drift inherent in open-loop prediction models. The CVAE method accomplishes this by producing a distribution over multiple possible trajectories at each timestep, which reduces drift that occurs due to multimodality (i.e. when only one plausible but incorrect mode of the future is predicted at a previous timestep, and used to predict the next timestep, other plausible and possibly correct modes are automatically ruled out, causing error which can accumulate when this process is repeated), as shown in an ablation study by Zhou et al. (2021). Meanwhile, as shown by Gopalakrishnan et al. (2019), the two-stage prediction method reduces drift by giving the model a future goal to focus and re-calibrate on, instead of merely a past history to propagate in a potentially erroneous direction.

This reduction of drift is also shown within the performance metrics in Table 6: compared to methods that don't use CVAEs or two-stage prediction, the ADE and FDE of the CVAE/two-stage modeling methods are a lot closer in value. For example, a majority of the methods which don't use these two techniques have metrics where the FDE is more than twice that of the ADE, whereas many of the methods that do use these techniques have an FDE that's much closer to their corresponding ADE. Since the ADE is an average distance error over the entire predicted trajectory, and therefore much more influenced by predictions in the near future than the FDE (final distance error), which is influenced by the longest term prediction in the far future, having an FDE which is closer in value to its corresponding ADE shows that the error accumulation over time is less.

Furthermore, the fact that most of the best performing CVAE/two-stage prediction methods don't use a map and/or a social modeling method, (although, as shown by Wang et al., 2022, adding a map encoding method further improves the results), shows even more strongly how well these two modeling techniques perform, even without this additional information. Moreover, this indicates that there is room for further improvement within these methods by adding robust map encoding and/or social interaction modeling techniques.

One interesting paper which uses almost the same exact methodology as Zhou et al. (2021), the best performing method in Table 6, is Mangalam et al. (2020b). Although

Mangalam et al. (2020b) also uses a CVAE and a two-stage predictor, and has the exact same social interaction modeling technique as Zhou et al. (2021), their CVAE is not recurrent: the CVAE is used only to predict the final future timestep (i.e. the 'goal'), and then a separate decoder network is used to produce a future trajectory out of the input, social interaction embedding, and 'goal' output by the CVAE. Therefore, the recurrent aspect of the CVAE within Zhou et al. (2021) may play a key role in its high performance. Although Mangalam et al. (2020b) was considered for inclusion in this survey, it was not included due to its similarity, in method, to Zhou et al. (2021), and because it reports worse performances than the top 5 methods in Table 6.

2.4 Discussion

In this section, we have discussed the input encoding methods, modeling methods, and training methods of the most common tasks within each of the prediction task categories presented. Across most tasks within action and motion prediction, there were a variety of input encoding methods that yielded high performances but there were a few modeling and training methods that seemed to perform better than others. Within modeling methods, the best performing methods seem to use GANs for tasks that predict a short time into the future, like early action prediction and RGB image generation, graph neural networks for tasks that predict slightly further into the future like action anticipation and human motion prediction, and CVAEs, as well as two-stage prediction methods for tasks that predict well into the future, like trajectory prediction. The trajectory prediction task discussed in this survey also uses a 'best-of-20' metric to encourage multimodal predictions, which the other prediction tasks discussed in this work do not consider to such an extent. Training methods vary by task, but common techniques are to add terms to the loss that encourage other objectives like goal detection, regularization, multitask learning, and smoothness of future predictions. Within trajectory prediction, however, adding KL divergence loss, and performing preliminary predictions that are later refined, shows significant improvements in performance.

3. Datasets

Although many surveys in the past have presented comprehensive lists of datasets (Rasouli, 2020; Rudenko et al., 2020) with details about environment, labels, and potential applications (Rasouli, 2020), many of these surveys organize datasets by year (Rasouli, 2020; Leon & Gavrilescu, 2021) or popularity (Rudenko et al., 2020). Such methods help to illustrate a history of the field, or point out datasets with the most papers to compare against, but are unwieldy for application-oriented readers unfamiliar with the field of prediction.

We aim to organize datasets such that someone with application-oriented goals can find the dataset that will best fit their problem requirements. Furthermore, unlike previous papers, we rate datasets on their level of social interaction, and highlight interaction focused and extremely long-tailed datasets.

To this end, we present Tables 7 and 8, which categorize datasets by their input modalities and labels, so that application constraints such as the availability of an ego camera or map can be taken into account. Tables 7 and 8 also describe environmental factors like types of agents present, filming location, and scene context to facilitate choosing

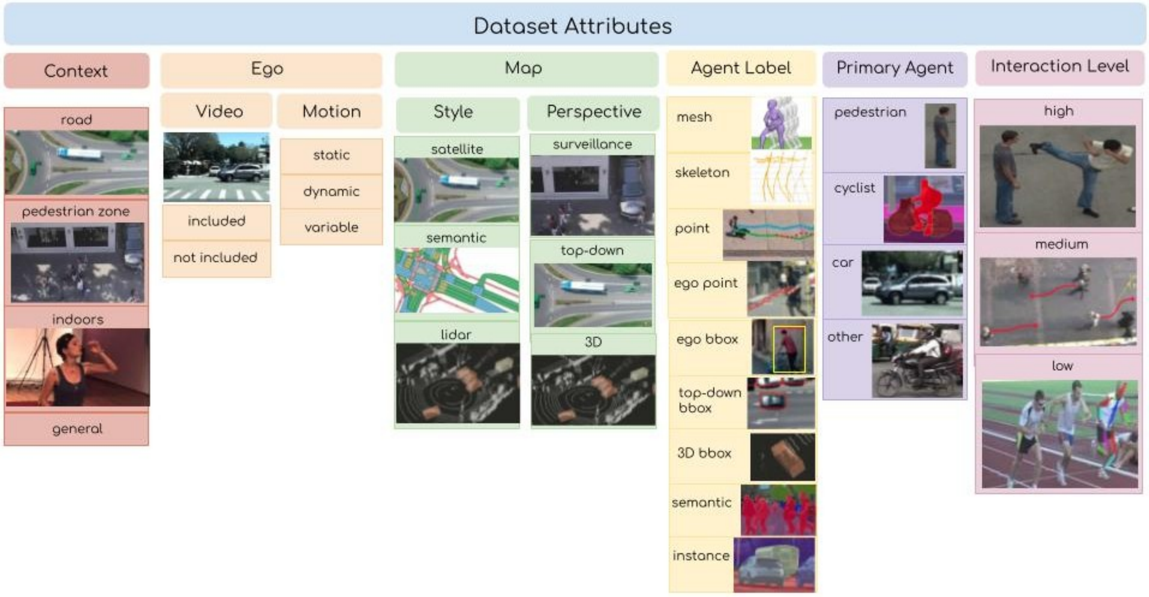

Figure 6: Illustration of the various dataset attributes used to summarize datasets in Tables 7 and 8. Images in this diagram are used for illustration purposes only and derived from Krajewski et al., 2020 (road, satellite); Pellegrini et al., 2010 (pedestrian zone, surveillance); Ionescu et al., 2014 (indoors); Caesar et al., 2020 (semantic); His, (lidar, 3D, 3D bbox); Yao et al., 2019 (video included, car); Adeli et al., 2021 (mesh); Wang et al., 2021 (skeleton); Ma et al., 2020 (point); Ped, (medium); UCB, 2018 (instance); Cordts et al., 2016 (semantic); Bock et al., 2019 (top-down bbox); Malla et al., 2020 (ego point); Chandra et al., 2019 (other); Ryoo & Aggarwal, 2010 (pedestrian, high); and Fig, (low). Please refer to Tables 7 and 8 to find the datasets that correspond to these attributes.

a dataset that's closest to the environmental conditions of a specific application. Finally, we detail the task types each dataset can be used for.

This selection of datasets aims to represent datasets that are widely used in literature, as well as highlight datasets that focus on social interaction and long-tailed learning. The novel taxonomy we propose in order to categorize and describe the datasets does not exhaust all possible modalities presented by all datasets, but instead identifies the most common ones in prediction (e.g. map types, agent label types, and ego camera). The following taxonomy describes the axes along which input modalities, environmental conditions, and dataset annotations are summarized in Tables 7 and 8:

1. **Context** indicates the type of environment in which the dataset was recorded. These include:
   (a) Road - paved roads for driving
   (b) Pedestrian zone - outdoor walking environments like universities and store fronts
   (c) Indoors - takes place within one or more rooms
   (d) General - contains all of the above environments and more
2. **Ego** is the point of view of an agent within the scene

   (a) Video - whether or not the dataset includes an ego perspective video
   (b) Motion - whether the camera of the ego agent or map satellite is static or dynamically moving through the scene (or variable by example)

3. **Map Style** indicates the type of map included. Options for map style include:
   (a) Semantic - semantically annotated with roads, sidewalks, etc.
   (b) Satellite - satellite style image taken of the area
   (c) Lidar - 3D point cloud of the area, but can be collapsed into one perspective (e.g. top-down) to create the map
4. **Map Perspective** is the point of view of the map (as opposed to ego, the point of view of the camera):
   (a) Surveillance - recorded from a structure overlooking the area from a slight angle
   (b) Top-down - perpendicular to the ground plane
   (c) 3D - data structure like a point cloud which can be viewed from any perspective in 3D (this implies that the map style is Lidar)
5. **Agent Label** is the way the agent is represented at each timestep. Options include:
   (a) Mesh - 3D mesh indicating a person's pose
   (b) Skeleton - skeletal representation of pose
   (c) Joints - joint locations relative to center of mass
   (d) Point - 2D coordinates of agent location from map perspective (in meters)
   (e) Ego Point - 2D coordinates of agent from ego video perspective (in pixels)
   (f) 3D bbox - bounding box coordinates in 3D
   (g) Ego bbox - 2D bounding box coordinates from ego perspective
   (h) Top-down bbox - 2D bounding box coordinates from map perspective
   (i) Instance - instance segmentation label
   (j) Semantic - semantic segmentation label
6. **Primary Agent** indicates the types of agents that are well represented by the dataset
7. **Location** is either a continent or media platform from which the data is recorded or sourced
8. **Task Type/Subtype** shows the task categories each dataset could be applied to. One caveat to the nomenclature, however, is that since action prediction datasets temporally crop the video around the action and can only be used for early action prediction, they are labeled as such, while datasets with uncropped videos (which can be used for both action subtasks) are labeled as anticipation.
9. **Interaction Level** indicate how much social interaction occurs in the dataset. Our method of quantifying this is described in Section 3.1.
10. **Duration** is the total duration of recorded frames
11. **No. Tracks** quantifies total number of agent tracks (sequences of frames in which the main agent can be seen continuously, which can be packaged as one example)

Context	Ego		Map		Agent Label	Primary Agent				Dataset
	Video	Motion	Style	Perspective		ped	cyc	car	other	
general	✓	static	-	-	mesh	✓				3DPW (von Marcard et al., 2018)
	✓	variable	-	-	skeleton	✓				MPII Human Pose (Andriluka et al., 2014)
	✓	variable	-	-	skeleton	✓				InstaVariety (Kanazawa et al., 2019)
	✓	variable	-	-	joints, ego bbox	✓				Penn Action (Zhang et al., 2013)
	✓	variable	-	-	joints	✓				JHMDB (Jhuang et al., 2013)
	✓	variable	-	-	skeleton	✓				UCF-101 (Soomro et al., 2012)
	✓	variable	-	-	-	✓				VideoLT (Zhang et al., 2021a)
	✓	variable	-	-	-	✓				TVSeries (De Geest et al., 2016)
	✓	variable	-	-	-	✓				TV-Human-Interaction (Patron-Perez et al., 2010)
	✓	variable	-	-	-	✓				Youtube-8M (Abu-El-Haija et al., 2016)
	✓	static	-	-	-	✓				BIT (Kong et al., 2012)
indoors	✓	static	-	-	joints, mesh	✓				Human3.6M (Ionescu et al., 2014)
pedestrian zone	✓	static	-	-	skeleton	✓				EgoPose (Yuan & Kitani, 2019)
		static	satellite	surveillance	point	✓				ETH (Pellegrini et al., 2010)
		static	satellite	surveillance	point	✓				UCY (Lerner et al., 2007)
		static	satellite	top-down	point	✓	✓		skater, cart	Stanford Drone (Robicquet et al., 2016)
		static	satellite	surveillance	point	✓				Trajnet (Sadeghian et al., 2018a)
	✓	static	-	-	-	✓				UTI (Ryoo & Aggarwal, 2010)
	✓	static	-	-	-	✓				KTH (Schuldt et al., 2004)
		static	satellite	surveillance	-	✓				ShanghaiTech Campus (Luo et al., 2017)
road	✓	dyn	-	-	3D bbox	✓	✓	✓		KITTI (Geiger et al., 2012)
	✓	dyn	-	-	ego bbox, instance	✓	✓	✓		BDD100K (Yu et al., 2020)

Continued on next page

Context	Ego		Map		Agent Label	Primary Agent				Dataset
	Video	Motion	Style	Perspective		ped	cyc	car	other	
road	✓	dyn	semantic	3D	3D bbox			✓		Argoverse (Chang et al., 2019)
	✓	dyn	semantic, satellite	3D	3D bbox			✓		Lyft L5 (Houston et al., 2020)
		static	satellite	top-down	top-down, bbox			✓		inD (Bock et al., 2020)
		static	semantic, satellite	top-down	point			✓		INTERACTION (Zhan et al., 2019)
	✓	dyn	-	-	ego bbox	✓	✓	✓	moped, rickshaw	TRAF (Chandra et al., 2019)
	✓	dyn	semantic	top-down	point, ego bbox	✓	✓			Euro-PVI (Bhattacharyya et al., 2021)
	✓	dyn	semantic	top-down	3D bbox			✓		nuScenes (Caesar et al., 2020) / PePScenes (Rasouli et al., 2020a)
	✓	dyn	lidar	top-down	3D bbox		✓	✓		LOKI (Girase et al., 2021)
	✓	dyn	-	-	ego bbox	✓				PIE (Rasouli et al., 2019)
	✓	dyn	-	-	ego point, ego bbox	✓		✓		TITAN (Malla et al., 2020)
	✓	dyn	-	-	ego bbox	✓				STIP (Liu et al., 2020)
	✓	dyn	-	-	ego bbox	✓				JAAD (Rasouli et al., 2017)
	✓	dyn	-	-	semantic	✓		✓		Cityscapes (Cordts et al., 2016)
	✓	dyn	-	-	ego bbox	✓				Caltech Pedestrians (Dollar et al., 2009)

Table 7: Table of **Relevant Datasets**. Summary of input modalities, environmental conditions, and dataset annotations of prediction datasets. Dashes indicate that the dataset does not include the corresponding annotations. More information about table columns can be found in Section 3. Abbreviations are: bbox, bounding box; dyn, dynamic.

Context	Task Type and Subtype			Duration	No. Tracks	Location	Interaction Level	Dataset
	Video	Action	Motion					
general	RGB		pose	51k frames	60	Europe		3DPW (von Marcard et al., 2018)
	RGB	anticipation	pose	24,920 frames	40,522	YouTube		MPII Human Pose (Andriluka et al., 2014)

Continued on next page

Context	Task Type and Subtype			Duration	No. Tracks	Location	Inter-action Level	Dataset
	Video	Action	Motion					
general	RGB		pose	24.3 hrs	-	Instagram	L	InstaVariety (Kanazawa et al., 2019)
	RGB	early action	pose	-	2,326	YouTube	L	Penn Action (Zhang et al., 2013)
	RGB	early action	pose	31838 frames	5,100	TV	L	JHMDB (Jhuang et al., 2013)
	RGB	antici-pation	pose	26.6 hrs	13,000	YouTube		UCF-101 (Soomro et al., 2012)
	RGB	antici-pation		-	256,218	YouTube		VideoLT (Zhang et al., 2021a)
	RGB	early action		16 hrs	6,231	TV		TVSeries (De Geest et al., 2016)
	RGB	antici-pation		-	400	TV	H	TV-Human-Interaction (Patron-Perez et al., 2010)
	RGB	early action		500k hrs	8,000,000	YouTube		Youtube-8M (Abu-El-Haija et al., 2016)
		early action		-	400	Asia	H	BIT (Kong et al., 2012)
indoors	RGB	early action	pose	450k frames	-	Europe	L	Human3.6M (Ionescu et al., 2014)
pedestrian zone	RGB		pose	8 mins	2	North America	L	EgoPose (Yuan & Kitani, 2019)
			traj	25 mins	650	Europe		ETH (Pellegrini et al., 2010)
			traj	16.5 mins	700	Europe		UCY (Lerner et al., 2007)
			traj	5 hrs	20,000	North America		Stanford Drone (Robicquet et al., 2016)
			traj	-	11,448	North America, Europe		Trajnet (Sadeghian et al., 2018a)
	RGB	early action		-	120	Asia	H	UTI (Ryoo & Aggarwal, 2010)
	RGB	early action		-	2,391	Europe	L	KTH (Schuldt et al., 2004)
	RGB			-	437	Asia		ShanghaiTech Campus (Luo et al., 2017)
road	RGB		traj	41k frames	-	Europe		KITTI (Geiger et al., 2012)
	sema-ntic		traj	318k bbox, 14k instance frames	131k bbox, 6.3k instance	North America		BDD100K (Yu et al., 2020)
	RGB		traj	1 hr	11,052	North America		Argoverse (Chang et al., 2019)
	RGB		traj	1,118 hrs	-	North America		Lyft L5 (Houston et al., 2020)
			traj	10 hrs	11,500	Europe	H	inD (Bock et al., 2020)

Continued on next page

Context	Task Type and Subtype			Duration	No. Tracks	Location	Interaction Level	Dataset
	Video	Action	Motion					
road			traj	10 hrs	40, 054	North America, Asia, Europe	H	INTERACTION (Zhan et al., 2019)
	RGB		traj	12.4k frames	246,512	Asia	H	TRAF (Chandra et al., 2019)
	RGB		traj	2.2 hrs	7,758	Europe	H	Euro-PVI (Bhattacharyya et al., 2021)
	RGB	anticipation	traj	5.5 hrs	17,081	North America, Asia	H	nuScenes (Caesar et al., 2020) / PePScenes (Rasouli et al., 2020a)
	RGB	anticipation	traj	2.25 hrs	28,000	Asia	H	LOKI (Girase et al., 2021)
	RGB	anticipation	traj	6 hrs	1,800	North America	H	PIE (Rasouli et al., 2019)
	RGB	anticipation	traj	10 hrs	14,096	North America	H	TITAN (Malla et al., 2020)
	RGB	anticipation	traj	2 hrs	3,348	North America	H	STIP (Liu et al., 2020)
	RGB	anticipation	traj	240 hrs	2,800	North America, Europe	H	JAAD (Rasouli et al., 2017)
	semantic			25k frames	-	Europe		Cityscapes (Cordts et al., 2016)
	RGB			250k frames	-	North America		Caltech Pedestrians (Dollar et al., 2009)

Table 8: Table of **Relevant Datasets**. Summary of environmental conditions, interaction level, dataset parameters, and task categories that prediction datasets can be applied to. Dashes indicate that the corresponding dataset parameter was not recorded in the literature. More information about table columns are found in Section 3. Abbreviations are: H, high; L, low; RGB, RGB Image Generation; semantic, semantic forecasting; anticipation, action anticipation; early action, early action prediction; pose, human pose prediction; traj, trajectory prediction.

3.1 Social Interaction in Datasets

Social interaction cues, which we define as the way in which multiple agents engage with each other and change the course of other agents' futures, can be learned explicitly to improve prediction methods by taking into account the influence of other agents. Typically, this is done through modeling techniques which are detailed in the following section, Section 4. However, in order for those modeling techniques to be able to capture and internalize the variety of social interactions that take place in the world, the datasets they are trained on must sufficiently sample such interactions. For example, using a small, staged dataset where most videos consist of one agent, wouldn't give the model as much opportunity to learn about social interaction as would using a large dataset collected on a public street, with 'interesting' scenes, like those with busy intersections, merging lanes, and lots of agents hand picked for labeling. Therefore, we create and include in this review (also included in

Table 8) a score for each of the datasets which describes the extent to which the dataset contains a variety of social interactions. In creating this score, we take into account the average number of people per scene, the location the data was taken from (e.g. data from straight highways has a lower score than data from multiway intersections or Indian traffic scenes where agents have more options and less regulation on which path to pursue), and whether there was any intentional curation done to increase the density of scenes with specific interactions (e.g. picking out scenes where there are interactions of a specific type and only labeling those like Kong et al., 2012 and Rasouli et al., 2019). One important aspect to choosing a dataset which we do not incorporate into our social interaction score is the size of the dataset, since the size can already be inferred from the 'Duration' and 'No. Tracks' columns in Table 8.

Therefore, all the datasets which are classified as high (H) interaction are those which were curated to increase the density of interactive scenes by either selecting for data with intersections, crosswalks, traffic, and other interaction events (Patron-Perez et al., 2010; Bock et al., 2020; Zhan et al., 2019; Chandra et al., 2019; Bhattacharyya et al., 2021; Caesar et al., 2020; Girase et al., 2021; Rasouli et al., 2019; Malla et al., 2020; Liu et al., 2020; Rasouli et al., 2017), or staged to include specific categories of interactions (Kong et al., 2012; Ryoo & Aggarwal, 2010). Meanwhile, all the datasets classified as low (L) interaction are either those whose scenes consist of only one agent (Jhuang et al., 2013; Ionescu et al., 2014; Yuan & Kitani, 2019; Schuldt et al., 2004), or those where only one agent in the scene is labeled (Zhang et al., 2013; Kanazawa et al., 2019). All other datasets are considered to have either average or unidentifiable levels of interaction (without deeper statistical analysis) and left blank.

While there are many Action Prediction datasets dedicated to learning social interaction (e.g., Girase et al., 2021; Rasouli et al., 2019; Malla et al., 2020; Liu et al., 2020; Rasouli et al., 2017; Patron-Perez et al., 2010; Kong et al., 2012; Ryoo & Aggarwal, 2010) and some Trajectory Prediction datasets that are starting to be curated that focus on social interaction (e.g., Girase et al., 2021; Rasouli et al., 2019; Malla et al., 2020; Liu et al., 2020; Rasouli et al., 2017) this area has yet to be explored in Human Pose Prediction and Video Prediction. One reason for this may be that these two task types don't typically predict farther than 0.5 to 1 second into the future, as shown in Tables 2 and 5. However, extending these tasks to be able to predict farther into the future is an important step towards developing more detailed future predictions, and developing models with an understanding of social interaction in these task types may aid in this endeavor.

3.2 Long Tailed Datasets

The only prediction dataset to our knowledge that's focused on long-tailed learning is the VideoLT (Zhang et al., 2021a) action prediction dataset. This dataset includes a large number of classes, and visually quantifies its own long tail as well as the long tails of other video classification datasets, as shown in Figure 3 of Zhang et al. (2021a).

However, not all datasets' long tails can be quantified as easily as action prediction datasets where class frequencies can be compared. Datasets which only contain trajectory prediction labels (e.g., Geiger et al., 2012; Sadeghian et al., 2018a; Yu et al., 2020; Chang et al., 2019; Houston et al., 2020; Bock et al., 2020; Zhan et al., 2019; Chandra et al., 2019;

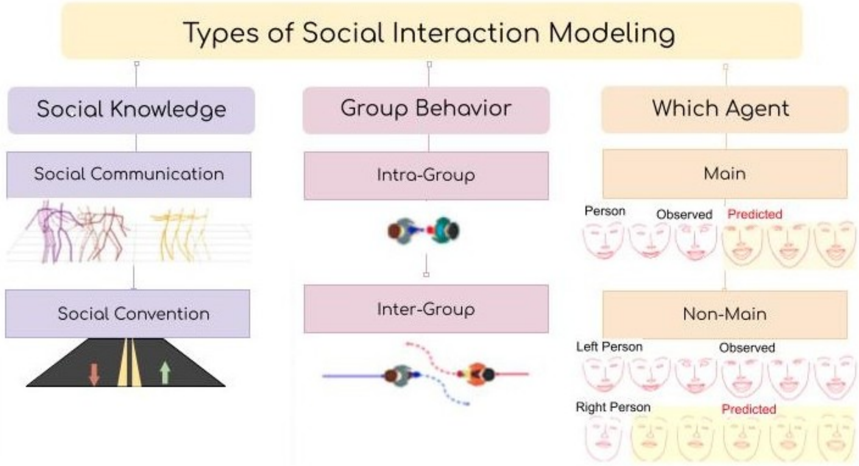

Figure 7: Types of Social Interactions that are modeled. Parts of this diagram are derived from Wang et al., 2021 (Social Communication diagram), Feng et al., 2017 (Main and Non-Main diagrams), and AIc, (Intra and Inter Group diagrams) and used for illustration purposes.

Bhattacharyya et al., 2021; Pellegrini et al., 2010; Lerner et al., 2007; Robicquet et al., 2016) or pose prediction labels (e.g., von Marcard et al., 2018; Yuan & Kitani, 2019) don't have a single measure like class frequency by which an example's rareness can be measured. A predicted pose sequence could involve rare postures, be moving at an uncommonly slow speed, or the skeleton itself could be rare, like when modeling an exceptionally small person.

Despite the difficulty in creating and quantifying a dataset that has a long tail across all relevant measures, the creation of such a dataset would have many benefits for evaluating and comparing separate long-tailed learning techniques. For example, since there is no one scale across which rareness can be measured, many algorithms (e.g., Makansi et al., 2021; Kozerawski et al., 2022) define and optimize for their own scale, making the results difficult to compare. However, with a dataset that's been curated to have a long tail across many different scales, methods optimizing for different scales can be compared. Therefore, in order to compare long-tailed motion prediction methods, we call for long-tailed datasets to be created within trajectory and pose prediction.

4. Social Interaction Modeling Techniques

One area of interest in current prediction methods is the development of modeling techniques that explicitly force the model to learn how multiple dynamic agents interact with each other and how that affects their future actions, trajectories, and expressions. These social interaction models typically attempt to better perform future prediction through the incorporation of model components that are dedicated to understanding the social relationships within the scene. In this survey, we focus on visible interactions indicated by kinesic cues like body positions, expressions, and actions and not verbal interactions.

To our knowledge, only one previous survey, Barquero et al. (2022), addresses the problem of non-verbal social interaction modeling within prediction, but this work only in-

Task Type	Sub-Task Type	Agent Type	Social Communication	Which Agent	Social Convention	Group Behavior	Interaction Modeling	Model
Video Prediction	Optical Flow	person	-	all	group	both	clustering	Cheriyadat and Radke (2008)
	Group Shape Prediction	person	-	all	group	both	3D-CNN	Wang and Steinfeld (2020)
	RGB Generation	person	expression	non-main	neural mirroring	intra	GAN	Huang and Khan (2017)
Action Prediction	Action Anticipation	person	classified interaction	all	-	intra	social pooling GAN	Airale et al. (2021)
			classified interaction	all	group	intra	AdaBoost	van Doorn (2018)
			gaze, classified interaction	all	group	both	attention	Sanghvi et al. (2020)
		vehicle	vehicle signal	main	-	both	ConvLSTM	Frossard et al. (2019)
			vehicle signal	main	-	both	spatiotemporal attention	Lee et al. (2019)
Motion Prediction	Trajectory Prediction	both	-	all	proxemics	both	tensor fusion	Zhao et al. (2019)
		person	-	-	avoidance	both	MLP	Wong et al. (2021)
			head	all	-	both	DiGraph LSTM	Zhang et al. (2019)
			head	all	-	both	Grid-LSTM	Haddad and Lam (2021)
			head	main	avoidance	inter	energy-based	Hasan et al. (2018)
			head	main	-	both	LSTM	Hasan et al. (2021)
			body	main	-	both	ConvLSTM	Chen et al. (2021)
			body	main	-	both	LSTM	Kao et al. (2021)
			body	all	-	both	3D-CNN	Kao and Chan (2022)
	Human Pose Prediction	person	body	all	-	both	GAT	Adeli et al. (2021)
			body	all	-	both	social pooling	Adeli et al. (2020)
			body	all	-	both	GAT	Corona et al. (2020)
			body	all	-	both	Transformer	Wang et al. (2021)
			body	all	-	intra	attention	Yasar and Iqbal (2021)
	Facial Pose Prediction	person	expression	all	neural mirroring	intra	VAE	Feng et al. (2017)
Other	Social Signal Inference	person	body	non-main	group	intra	VAE	Joo et al. (2019)
	Face Embedding	person	expression	main	-	both	VAE	Wiles et al. (2018)
	Relationship Inference	person	expression	main	-	both	CNN	Zhang et al. (2017)

Table 9: Summary of **Social Interaction Modeling Techniques** within prediction and other relevant tasks, described by the type of social interaction, and modeling architecture. Details about social interaction categories can be found in Section 4. Abbreviations include: DiGraph, directed graph; Grid-LSTM, LSTM cells organized in a grid; VAE, variational autoencoder; avoidance, collision avoidance; group, group dynamics; head, head position; body, body pose.

cludes 'focused' interactions, where 'focused' indicates continuous social interaction between agents, including two-person conversations, but not trajectory prediction. Furthermore, the social forecasting taxonomy presented in Barquero et al. (2022) is extremely general, not specific to social behavior.

Since trajectory prediction is one of the main areas of prediction where non-verbal social interaction modules are making an improvement, as is evidenced by the overwhelming number of social interaction based trajectory prediction methods (e.g., Sadeghian et al., 2018b; Hasan et al., 2018; Huang et al., 2019; Zhao et al., 2019; Kosaraju et al., 2019; Salzmann et al., 2021; Wong et al., 2021; Zhou et al., 2021; Zhang et al., 2019; Haddad & Lam, 2021; Chen et al., 2021; Kao et al., 2021), we believe this topic to be key in any discussion about visual social interaction modules within prediction. Therefore, we review both 'focused' and unfocused non-verbal social interaction prediction methods. The set of methods described here and summarized in Table 9 are chosen to display the variety in interaction modeling methods, and in order to highlight important breakthroughs in visual social modeling.

4.1 Taxonomy

In order to systematically summarize the social interaction methods surveyed, we propose the following taxonomy, which is further illustrated in Figure 7. In this taxonomy, we categorize each method by *what* kinds of scenarios are observed, *who* in the scene is modeled, and *how* these visible interactions are made known to others. Within the methods surveyed, we found that the types of scenarios observed typically fell into the overarching categories of Intra-Group scenarios (observing the minute details of those conversing within a group), Inter-Group scenarios (observing how the macro paths of individuals are influenced by those they are not involved with), and Group-Agnostic (methods which can be applied to both Intra and Inter-Group scenarios). Similarly, the person whose interaction is being modeled is typically either the person whose future is being predicted (Main agent), someone interacting with the main agent (Non-main agent), or all the agents in the scene. Finally, we found that the way that these visual interactions are made known to others is either through explicit cues such as hand or head gestures, or through implicit knowledge of conventions such as staying on one side of the road.

Social Knowledge: How are visual social interactions made known to others?

1. **Social Communication** - agents indicate their future path via explicit kinesic cues
2. **Social Convention** - agents follow pre-programmed social norms that are implicitly known without need for explicit cues

Group Behavior: What kinds of social scenarios are observed?

1. **Intra-Group** - visual interactions between members of a group, like expression mirroring, inter-personal distance, etc.
2. **Inter-Group** - visual interactions between members of different groups, like collision-avoidance, right-of-way behavior, etc.
3. **Group-Agnostic** - interactions like gesturing, looking where you're going, etc. that indicate future motion independent of group identity

Which Agent: Whose social signals are modeled?

1. **Main** - the agent whose future is being predicted is the one whose social signals are modeled
2. **Non-Main** - the agent interacting most with the main agent (whose future is predicted) is the one whose social signals are modeled
3. **All** - social signals of all agents in the scene are modeled and one or all agents' futures are predicted

Social Knowledge is the way in which the knowledge of how to interact is gained by the agents. Some interaction dynamics, such as keeping distance from those in other groups, or following those within one's group, are implicit knowledge that does not have to be communicated in an explicit way (Hasan et al., 2018; Wong et al., 2021; Zhao et al., 2019; Cheriyadat & Radke, 2008; Wang & Steinfeld, 2020). However, other interaction cues, such as signaling in which direction one is headed or deducing another's destination by where they are looking, are explicitly communicated (Zhang et al., 2019; Haddad & Lam, 2021; Hasan et al., 2021; Adeli et al., 2021; Chen et al., 2021; Kao & Chan, 2022; Corona et al., 2020; Wang et al., 2021; Yasar & Iqbal, 2021).

Group Behavior. One way to categorize what scenario a visual social interaction method is focused on is by whether it models close-up intra-group behaviors (like the distance kept between individuals talking in a circle, as modeled by Hung & Krose, 2011, or how expressions are mirrored between two people in a conversation, as modeled by Feng et al., 2017), or larger inter-group behaviors (like the way in which two groups pass each other, as modeled by Wang & Steinfeld, 2020, or how two people going separate ways avoid collision, as studied by Hasan et al., 2018). However, many social modeling methods, like those that model changes in body pose (e.g., Chen et al., 2021; Kao et al., 2021; Kao & Chan, 2022), are applicable to both inter-group and intra-group scenarios.

Which Agent. Finally, we also categorize social interactions based on whose interactions are being modeled compared to whose future is being predicted. We use *main* to indicate the agent whose future is being predicted, *non-main* to indicate the single agent who is most interacting with the main agent, and *all* to indicate all agents in the scene. Therefore, we can group prediction algorithms into those that predict the action of the main agent by looking at the social signals exhibited by only the main agent (e.g., Frossard et al., 2019; Lee et al., 2019; Hasan et al., 2018, 2021; Chen et al., 2021; Kao et al., 2021), those that predict the action of the main agent by only looking at the non-main agent (e.g., Huang & Khan, 2017; Joo et al., 2019), those that predict the action of the main agent by looking at the actions of all agents (e.g., Corona et al., 2020; Wang et al., 2021; Feng et al., 2017; van Doorn, 2018; Sanghvi et al., 2020), and those that predict the action of all agents by looking at all agents (e.g., Kao & Chan, 2022; Adeli et al., 2021, 2020; Yasar & Iqbal, 2021; Zhang et al., 2019; Haddad & Lam, 2021; Zhao et al., 2019; Airale et al., 2021; Cheriyadat & Radke, 2008; Wang & Steinfeld, 2020). Many of the methods that use solely non-main agent kinesics were developed for human motion imitation tasks such as training a robot to imitate the expressions of a seller given that of a buyer.

4.2 Summary of Interaction Models

We summarize the various interaction modeling methods relevant to the field of prediction by indicating how each method answers the above three questions in Table 9. In the table, the questions of which agent's social interaction is being modeled and which type of group scenario the focus is on, are specified under the 'Which Agent' and 'Group Behavior' columns, respectively. However, for the social knowledge question, many of the methods focus on modeling multiple types of social communication and/or convention so we have added separate 'Social Communication' and 'Social Convention' columns, each of which further categorizes the types of social communication and/or convention modeled by each method, as shown in Table 9. Finally, we also summarize the type of modeling method used to capture the social interaction under the 'Interaction Modeling' column.

In this section, instead of concentrating on methods shown to do well in open-ended environments as we have been, we expand our horizon to include methods studied in focused environments (like those discussed in Barquero et al., 2022) to promote comparison between works within a diverse range of applications. Furthermore, in many instances within action prediction, non-verbal social interactions are learned through interaction-only action prediction datasets like UT-Interaction (Ryoo & Aggarwal, 2010) and BIT-Interaction (Kong et al., 2012) instead of through modeling techniques. While these datasets, which are presented in Section 3.1, help models learn social interactions, they are not social interaction modeling *techniques* and are therefore not discussed in this section.

In the following subsections, we review in more depth each of the non-verbal social interaction modeling techniques summarized in Table 9. As most of these methods directly model or incorporate into their loss the various types of *social knowledge* cues on which they focus, while the *group behavior* and *which agent* categories are typically more related to the formulation of the problem than the social modeling methodology, we will organize this review by grouping the methods by the types of social communication and/or convention cues they model, and comparing techniques which model the same type of social knowledge cue.

4.2.1 Social Communication

Expression. In Huang and Khan (2017), the kinesic cue of a person's expressions can be used to predict those of another using two GANs: one to produce a sketch of the non-main agent from their facial features, and another to infer the facial image of the main agent from the sketch. Similarly, Feng et al. (2017), predict the facial pose (locations of facial feature markers like eyes, nose, etc.) of an agent in two-person interactions by using a variational autoencoder on the past facial poses of both agents. Other methods, like Wiles et al. (2018) use unsupervised variational autoencoders to learn face embeddings which may be applied to incorporate expressions into prediction algorithms. Meanwhile, Zhang et al. (2017) predict interpersonal relationships which can be used to inform multi-agent prediction by extracting visual features from every pair of faces, generating pseudo attribute labels for each person, and combining them with other spatial cues to predict the relation between the pair.

Body crops, or images cropped to the size of bounding boxes around a person's body, are used in early action prediction (e.g., Wu et al., 2021), and action anticipation (e.g., Yao et al., 2021) to create relational scene graphs that encapsulate the relationships between actors and objects in a scene. These methods first perform object detection, and then feed the detections through an attentive relation network (ARN) like in Yao et al. (2021), or long short term graph convolutional network (LST-GCN) like in Wu et al. (2021) to tease out the relationships between the actors and objects.

Classified Interaction. Another type of kinesic cue is the sequence of past interactive actions (classified interactions), which can help determine the future interactive action sequence. Airale et al. (2021) model this with a GAN where historical actions are encoded via LSTM, while van Doorn (2018) use agents' past locations and action kinesics like whether they're stepping or laughing to predict which agents will leave a conversation group. Sanghvi et al. (2020), on the other hand, uses past positions, gaze directions, and classified interactions of all agents in the scene to predict future actions via an attention-based short-term memory encoder.

Vehicle Signals. While most social interaction methods focus on pedestrians, vehicles can also indicate their intentions through *vehicle signals* like turn signals and brake lights, which we consider to be vehicle kinesics. As there is no work on incorporating vehicle signal detection into a trajectory prediction framework, we present the following vehicle signal detection methods as pseudo-prediction methods and ignore situations with misleading signals. Frossard et al. (2019) classifies visual and temporal features extracted from a convolutional LSTM (ConvLSTM), while Lee et al. (2019) applies spatial and temporal attention on frame-to-frame optical flow maps to detect the signal.

Head/Body Pose. Many works in motion prediction either use head orientation (e.g., Zhang et al., 2019; Haddad & Lam, 2021; Hasan et al., 2018, 2021) or body position (e.g., Chen et al., 2021; Kao et al., 2021; Kao & Chan, 2022; Adeli et al., 2021, 2020; Corona et al., 2020; Wang et al., 2021; Yasar & Iqbal, 2021) to predict the future trajectory or pose of the main agent.

Since all human pose prediction algorithms inherently use the main agent's body pose history (as covered in Section 2.3.1), we only consider pose prediction algorithms which incorporate poses of all agents in the scene to be social interaction models. Social interaction modeling techniques used in pose prediction problems which incorporate body poses of all other agents in the scene include: social pooling, which pools features across embeddings of all agents (e.g., Adeli et al., 2020), Transformer networks that use an encoding of the main agent and all agents as key and value, respectively (e.g., Wang et al., 2021), cross-agent attention (e.g., Yasar & Iqbal, 2021), and graph attention networks that use attention on both human-human, and human-object interactions (e.g., Adeli et al., 2021; Corona et al., 2020).

In trajectory prediction, body posture can be incorporated via a simple 'head, hands, and feet' positional model where either the posture of the main agent (e.g., Kao et al., 2021) or the postures of all agents (e.g., Kao & Chan, 2022) over time are input into a simple LSTM network (e.g., Kao et al., 2021) or 3D convolutional network (e.g., Kao & Chan, 2022) to accumulate pose information. Chen et al. (2021), however, concatenates the main agent's postural skeleton history with spatial features and feeds it through a convolutional LSTM network with an attention module in order to force the prediction to rely on more than just the previous timestep. While Hasan et al. (2021) and Hasan et al. (2018) use head orientation to remove pedestrians outside the field of view of the main agent, Zhang et al. (2019) form directed graphs of all the pedestrians in the scene based on who can see who, and Haddad and Lam (2021) encode the head orientation of all agents as 'vislets', combine it with both spatial and historical features separately via two grid-LSTMs, and combine them both using a Gated Graph Recurrent Neighborhood Network.

4.2.2 Social Convention

Group Dynamics involves modeling the way groups move and shift within the environment. For example, Cheriyadat and Radke (2008) identifies optical flow patterns within groups moving through crowded metro stations by clustering movements and projecting them into the future. Meanwhile, Wang and Steinfeld (2020) uses videos with semantically segmented blobs tracking group movement, and predicts the future shape of the group by using a 3D CNN on the spatiotemporal cube. Finally, van Doorn (2018) predicts whether an agent will leave the group, while Sanghvi et al. (2020) and Joo et al. (2019) predict future actions and body positions, respectively, of only the non-main agent on datasets that center around single social groups like small parties (e.g., Sanghvi et al., 2020) or three-person negotiations (e.g., Joo et al., 2019).

Proxemics and Collision Avoidance. Proxemics involves modeling the optimal space between one person and another (which may vary by culture and location), while collision avoidance is the path planning that agents perform in order to 1) avoid chances of collision with another person, and 2) indicate to the other person through early changes in their path how they intend to prevent collision. While Hasan et al. (2018) incorporate collision avoidance into their model of the future by explicitly including a term in the loss function that encourages larger distances between the predicted paths of the main agent and the agent closest, Wong et al. (2021) creates a two-stage coarse and fine prediction system where the fine system learns to interpolate between and maneuver the points on the coarse path such that agents don't collide. To model proxemics, Zhao et al. (2019) embeds each agent's encoded historical information onto a map, thereby implicitly learning the proxemics that agents prefer to keep.

Neural Mirroring. Finally, neural mirroring is the inclination to mirror the faces of those we are in conversation with due to empathetic neural pathways. Some works attempt to predict the future expressions (represented as facial images or facial poses) of the main agent by looking at past images of the non-main agent (e.g., Huang & Khan, 2017) or facial poses of both agents (e.g., Feng et al., 2017).

5. Long-Tailed Learning Techniques

One challenge of open-ended environments is that behaviors encountered in such real-world scenes resemble a long tailed distribution. There are many examples of easily predictable behaviors like standing still or traveling at a constant velocity, and few examples of complicated behaviors like stopping to tie a shoelace. In this section, we discuss how to overcome such dataset imbalances in classification and regression problems, both within and outside of prediction, and summarize the methods in Table 10.

Many classification surveys have covered the plethora of long-tailed learning techniques within the various classification problems of image recognition (e.g., Zhang et al., 2021b), action recognition (e.g., Özyer et al., 2021; Vrigkas et al., 2015; Yadav et al., 2021), semantic segmentation (e.g., Jadon, 2020; Sampath et al., 2021), and action prediction (e.g., Rasouli et al., 2020b; Xu et al., 2020; Rasouli et al., 2020a; Zaech et al., 2020). There

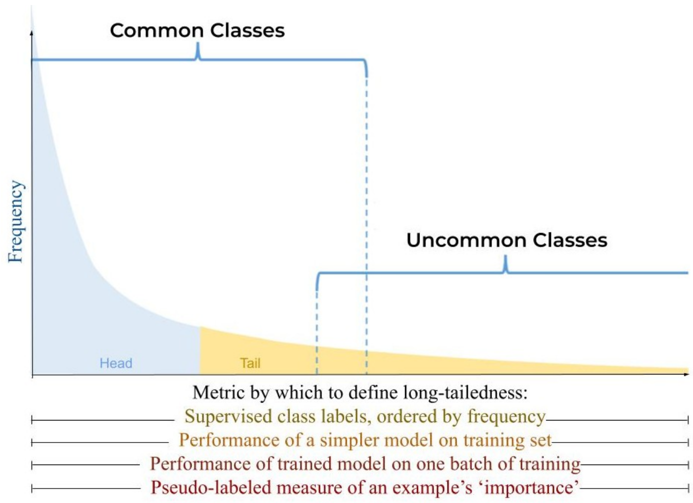
Common Classes
Uncommon Classes
Frequency
Head
Tail
Metric by which to define long-tailedness:
Supervised class labels, ordered by frequency
Performance of a simpler model on training set
Performance of trained model on one batch of training
Pseudo-labeled measure of an example's 'importance'

Figure 8: Ways to define a long-tailed distribution. Parts of this diagram are derived from Liu et al. (2019).

are also classification datasets specifically curated for long-tailed recognition of images (e.g., Liu et al., 2019) and videos (e.g., Zhang et al., 2021a).

However, dealing with imbalanced datasets in regression is more complicated, especially in multidimensional regression tasks like video and motion prediction, because defining a metric by which to determine whether an example falls into the long tail is non-trivial. In classification it is straightforward to categorize a dataset as imbalanced by showing that there are many examples in one class and very few examples in another, but in order to do this for multi-dimensional regression tasks, there has to be a scale on which examples are compared. Are they imbalanced in terms of the lengths of their trajectories, average velocities, or the level of social interaction an agent engages in? The answer, typically, is that in open-ended environments, datasets are imbalanced on many different scales (Oksuz et al., 2021). Furthermore, any one of these metrics is not enough to adequately capture the long-tailed nature of the problem: if we define imbalance along the scale of velocity, we may fail to highlight examples with uncommon interaction levels. On the other hand, these metrics may be related: a pedestrian who is walking with a friend may be more likely to walk slower and at the same pace as them. Analyzing the co-occurence of the different possible metrics and developing a scale on which to define imbalance is an important and non-trivial future problem in multi-dimensional regression.

Long-tailed learning in single-value regression (e.g., Branco et al., 2017; Moniz et al., 2017; Steininger et al., 2021) is slightly easier as there is a single scale along which imbalance can be measured, but it is still non-trivial to extend the well-studied class-based long-tailed learning techniques to any type of regression.

Since previous surveys like Zhang et al. (2021b) have sufficiently covered the plethora of long-tailed learning techniques for classification tasks outside of prediction, we will first review these techniques by briefly summarizing the taxonomy presented in Zhang et al. (2021b), and use it more generally to describe the work done on long-tailed learning within

Task Type	Task Subtype	Long-Tail Learning			Model
		Technique Category	Technique Details	Class Definition	
Action	anticipation	re-sampling	combining classes	class label	(Xu et al., 2020)
		cost-sensitive	re-weighting	class label	(Rasouli et al., 2020a)
		cost-sensitive	re-weighting	class label	(Rasouli et al., 2020b)
		cost-sensitive	re-weighting	class label	(Zaech et al., 2020)
Motion	pose	transfer learning	pre-train on common classes	class label	(Zang et al., 2020)
	trajectory	representation	supervised contrastive learning	performance-based	(Makansi et al., 2021)
		cost-sensitive	re-weighting	performance-based	(Kozerawski et al., 2022)
		representation	metric learning	-	(Li et al., 2021b)
		data augmentation	stochastic simulation	-	(Anderson et al., 2019)
Other	regression	re-sampling + data augmentation	undersampling + interpolation	importance tag	(Branco et al., 2017)
		ensemble learning	boosting/ bagging	importance tag	(Moniz et al., 2017)
		cost-sensitive	re-weighting	-	(Steininger et al., 2021)
	image depth estimation	cost-sensitive + classifier design	re-weighting + embedded feature smoothing	-	(Yang et al., 2021)
	human mesh recovery	logit adjustment	softmax with temperature	batch performance-based	(Ren et al., 2022)

Table 10: Summary of **Long-Tailed Learning Techniques** within prediction and regression, categorized by the long-tailed learning categories defined in Zhang et al. (2021b) and summarized in Section 5.1. *Technique details* describe the specific technique within the category. *Class definition* indicates whether or not classes are defined, and specifies how they were derived: class label indicates a supervised label, while performance-based and importance tag class definitions are self-supervised. Abbreviations are: regression, single-value regression; re-weighting, weighting the loss proportional to class infrequency.

regression and prediction, as shown in Table 10. To our knowledge, we are the first work to survey long-tailed learning techniques outside of classification.

5.1 Classification

5.1.1 Classification Taxonomy

Zhang et al. (2021b) summarized the many long-tailed learning techniques within classification, and organized them into the following categories:

1. Class Re-balancing Methods:
   (a) Re-sampling
   (b) Cost-sensitive learning
   (c) Logit adjustment
2. Information Augmentation Methods:
   (a) Transfer learning
   (b) Data augmentation
3. Module Improvement Methods
   (a) Representation learning
   (b) Classifier design
   (c) Decoupled training
   (d) Ensemble learning

In the following paragraphs, we provide more details about this taxonomy by summarizing the definitions and boundaries proposed by Zhang et al. (2021b) for each of the categories and subcategories within the taxonomy.

Class Re-balancing Methods attempt to effectively force the class frequencies to be equal by manipulating the data, loss, or output logits. *Re-sampling* oversamples (by duplicating) examples from uncommon classes and undersamples examples from common classes to alter the class distribution at the expense of losing much of the data. This method does not perform as well on extremely skewed datasets as extreme oversampling causes overfitting of tail classes and extreme undersampling degrades performance on head classes. *Cost sensitive Learning* adds weight to the loss of rare examples such that the weight is proportional to measures of 'rareness' such as inverse class frequency or class prediction 'hardness' (as used in focal loss from Lin et al., 2018). Although many of these methods are theoretically inspired, they also suffer from overfitting to tail classes but to less of an extent than the re-sampling methods. *Cost sensitive Learning* changes output class scores post-hoc based on label frequencies by boosting the class scores with low label frequency. Although class re-balancing methods are generally simple to implement and show performance improvements, they essentially can't handle the issue of lacking information due to limited data so improving tail performance typically involves the trade-off of also regressing head performance to some extent.

Information Augmentation Methods introduce additional information into the training from other sources in order to improve tail performance without sacrificing head performance. For example, *transfer learning* transfers knowledge from a source domain such as other datasets, tasks, or classes, to improve performance on a target domain (Zhang et al., 2021b). This includes head-to-tail knowledge transfer, where class-agnostic features like intraclass variance of head classes are used to guide feature augmentation for tail-class samples so that the tail-class features have higher intraclass variance and other class-agnostic features similar to head classes. Another transfer learning technique is to pre-train the model by either first training only on the tail classes, pre-training on a self-supervised network like contrastive learning, or pre-training with an additional modality like a language model. Knowledge distillation methods like training a smaller model off the outputs of a bigger model (student-teacher model), and self-training methods that use a large unlabeled dataset in addition to the smaller labeled one such that the large unlabeled dataset can be pseudo-labeled by the model and used to increase the class frequency of rare classes, also fall under this category. *Data augmentation* methods, on the other hand, create new examples through class-conditional statistics (e.g. generating class-wise features based on a learned Gaussian prior with its mean and variance estimated from previously observed samples like in Anderson et al., 2019) in order to synthesize examples from uncommon classes (Zhang et al., 2021b). These methods improve tail performance without sacrificing head performance by learning or incorporating additional information.

Module Improvement Methods change the network structure such that uncommon examples perform better without sacrificing the performance of common examples. *Representation learning* does this by manipulating the loss function to change the feature space in such a way that pushes apart the feature embeddings of examples from different groups and pulls together examples within the same group (e.g. supervised contrastive learning techniques like the 'range loss' in Zhang et al., 2016). These groups can be the individual classes, or sets of classes which fall under categorizations such as 'common' vs. 'uncommon.' Some representation learning methods additionally employ memory modules where visual features of each class are stored in an independent memory block which is uninfluenced by that class's frequency (e.g. Liu et al., 2019). Meanwhile, some methods employ *classifier design* techniques such as causal modeling (which records the bias by computing the exponential moving average of features during training, and then removes the bias by subtracting the bias from prediction logits during inference), nearest-neighbors based approaches (which predict the class whose mean feature embedding is closest to that of the example in question), and non-linear regularization (which weights examples using a learnable parameter which uses regression to figure out how much bias should be applied towards rare examples), as explained further in Zhang et al., 2021b. Another way in which modules can be improved to reduce the effects of class imbalance is through decoupled training and ensemble learning techniques. *Decoupled training*, which separates the learning procedure into representation learning and classifier training stages to be individually evaluated, can be done by introducing data mix-ups solely within the representation learning stage (as this was shown to be beneficial for representation learning but detrimental for classification), or improving the classification by minimizing the KL-Divergence between the calibrated prediction distribution and a balanced reference distribution (Zhang et al., 2021b). Finally, *ensemble learning* techniques strategically generate and combine multiple network modules specializing in different aspects of the data/modeling. While some networks use an extra network branch which only trains on tail classes and combines the outputs dynamically during training, others split the data into subgroups by dividing the long-tailed dataset into several subsets with less dataset imbalance (Zhang et al., 2021b). Some even add a term to the loss that forces the model to improve the diversity of the different experts. Although representation learning and classifier design methods can improve results, they still suffer from overfitting due to lack of added information. Decoupled training and ensemble learning methods, on the other hand, rarely see a drop in performance on head classes but they incur higher computational costs. Therefore, if model efficiency is a priority, using representation learning and classifier design techniques would be preferable to implementing decoupled training and ensemble learning.

In the following subsections, we use this taxonomy to categorize long-tailed learning methods in action prediction, video and motion prediction, and other regression methods outside of prediction. Although this taxonomy was developed for classification techniques, the approaches used in regression problems typically have analogous methods within classification (Ren et al., 2022), and the categories are broad enough to represent different ways of thinking about the long tail problem. Therefore, we find that this taxonomy can be used to describe long-tailed learning techniques within regression as well.

5.1.2 Action Prediction

Although there are many ways to overcome class imbalance in classification problems like image recognition (Zhang et al., 2021b), and in theory, most of these methods should be applicable to action prediction tasks as well, the majority of these methods have not yet been used in works regarding action prediction. Of the taxonomy of long-tailed learning methods presented in Zhang et al. (2021b), only those of the class re-balancing category have thus far been applied to action prediction, with one work employing re-sampling by combining multiple rare classes into one class (Xu et al., 2020) in order to eliminate the long tail and increase the size of the one 'rare' class, and some works using cost-sensitive learning by re-weighting samples by the ratio of their positive to negative samples in order to add weight to rare classes that have fewer positive samples (e.g., Rasouli et al., 2020a, 2020b). Zaech et al. (2020), on the other hand, re-weights samples by the presence or lack of interesting events like lane changes and turns, with the assumption that interesting events are more rare, but more important to learn, and therefore deserve more weight. One promising research direction may be to apply long-tailed learning techniques to action prediction and evaluate them on the VideoLT (Zhang et al., 2021a) dataset, which quantifies its long tail and provides metrics for long tailed performance.

5.2 Regression

5.2.1 Video and Motion Prediction

Despite the relative lack of works exploring dataset imbalance in action prediction, there are even fewer works in video prediction. At the time of writing, we've encountered a complete lack of papers on long-tailed learning in video prediction.

Within motion prediction, Makansi et al. (2021) and Kozerawski et al. (2022) directly address long-tailed learning in trajectory prediction, while Li et al. (2021b) simply show that injecting logic rules by adding cross-walks, traffic lights, and left/right turn only lanes into the map and making them hard rules instead of suggestions to be used as input, reduces the long tail of the error distribution, as shown in Figure 3 of Li et al. (2021b). Although Anderson et al. (2019) don't directly address dataset imbalance, they develop a data augmentation method that could be used to upsample uncommon trajectories by generating trajectories from dataset statistics and adding random transformations to increase the variety and number of trajectories.

To improve long-tail performance, Makansi et al. (2021) use contrastive loss on implicit classes of trajectories to force the model to learn the characteristics of rare trajectories separately from common trajectories. This loss forces the feature embeddings of the rare trajectories to be pushed apart from the feature embeddings of common trajectories, in the feature space (Makansi et al., 2021). Therefore, feature embeddings of rare trajectories are less likely to be lost within the manifold of common trajectories, and assumed to be outliers. In Makansi et al. (2021), classes are defined by how easy it is to predict the future trajectory through a physics-based Kalman filter: rare and important trajectories are assumed to be the ones which are difficult to predict using simple kinematics. Kozerawski et al. (2022), on the other hand, compare two novel loss terms that up-weight rare, high error examples: a regularization term which improves performance slightly in average and rare cases, and a kurtosis term which significantly improves only the worst error. The regularization term includes hyperparameters that assume a fixed shape for the error distribution while the kurtosis term uses batch statistics to estimate the error distribution. While Makansi et al. (2021) and Kozerawski et al. (2022) show small improvements in averaged metrics, it is difficult to compare improvements in the long tail as each paper defines and optimizes for their own scale of uncommonness. Therefore, a standardized method of evaluating long tailed performance must be defined for trajectory prediction.

Finally, Zang et al. (2020), the only work that explicitly addresses class imbalance in human pose prediction, uses action class labels from the Human3.6M (Ionescu et al., 2014) dataset to create a class imbalanced dataset. Common classes are used for pre-training and uncommon classes are learned through a parameter generation module that adapts the model to new categories (Zang et al., 2020).

5.2.2 Other Regression

Outside of prediction, there are also several single-dimensional (e.g., Branco et al., 2017; Moniz et al., 2017; Steininger et al., 2021) and multi-dimensional (e.g., Yang et al., 2021; Ren et al., 2022) regression tasks that incorporate long-tailed learning.

Some regression methods take advantage of long-tailed classification techniques by defining pseudo-class labels through unsupervised classification of regression examples (e.g., Branco et al., 2017; Moniz et al., 2017; Ren et al., 2022). Branco et al. (2017) and Moniz et al. (2017) classify examples by calculating a domain-specific importance tag for each example, where extreme values (possibly including outliers) are considered 'important'. Branco et al. (2017) then performs re-sampling by undersampling 'unimportant' examples, and data-augmenting 'important' examples by using k-nearest-neighbors based intorpolation between examples, while Moniz et al. (2017) performs ensemble learning by *boosting* (using multiple models) and *bagging* (using multiple sets of bootstrapped samples from the training set), with performance averaged across the 'boosted' models and 'bagged' sets. Boosting and bagging helps train multiple models that specialize on different parts of the dataset, which helps rare examples get identified and predicted with their own specialized model (Moniz et al., 2017). Ren et al. (2022), on the other hand, classify examples using performance-based labels where examples that perform similarly within a batch fall into the same class. They then adjust the loss by adding weight to the worst performing examples in order to improve prediction of the lowest performing long tail. One disadvantage of the above techniques, however, is the heuristic division of the dataset into rare and frequent sets (Yang et al., 2021). This causes a tendency to classify outliers as important, rare examples. Especially in applications with noisy data, amplifying genuine outliers along with the long tail produces improved performance in the long tail, at the expense of average performance (Moniz et al., 2017).

Other methods use techniques like label and feature smoothing instead of defining pseudo-labels in order to take advantage of the fact that in most regression tasks, there is a dependence between data samples which have nearby labels (e.g., facial images of close ages) such that data from nearby labels can be used to boost the learning of the more sparsely represented data points in between (Yang et al., 2021). Furthermore, if the model works properly and the data is balanced, one expects the feature statistics corresponding to nearby targets to be close to each other, producing continuity in feature space (Yang et al., 2021). In these works, the distribution of regression labels across all values is smoothed via kernel density estimation (e.g., Steininger et al., 2021) or another type of kernel convolution (e.g., Yang et al., 2021), and examples are weighted by the inverse of their label's smoothed frequency, since error distribution is shown to inversely correlate with label density distribution (Yang et al., 2021). The kernel convolution developed by Yang et al. (2021) specifically produces the expected density of the real (smoothed) label distribution, which accounts for the information which can be learned from examples with nearby labels, and can even be applied to multi-dimensional inputs like in image depth estimation by treating the depth of each pixel as a label to be smoothed and weighted (Yang et al., 2021). Since continuity in the target space should create a corresponding continuity in the feature space , Yang et al. (2021) uses an additional 'classifier' (or in this case, 'regressor') design technique to perform feature space smoothing between neighboring examples of the label distribution, as shown in Figure 4 of Yang et al. (2021). These methods have a lower chance of suffering from the noise outlier problem mentioned above.

Applications to Prediction. Although some of these techniques are specific to one dimensional regression tasks, others can be easily applicable to video and motion prediction. For example, Ren et al. (2022) focuses on dataset imbalance in human mesh recovery, a multi-dimensional regression task which could work on trajectory or human pose prediction because of similarities in output formats. Similarly, Yang et al. (2021), which studies image depth estimation, could be applied to the video prediction subtask of optical flow map generation, as both are generative tasks with semantically meaningful pixel-level labels. Therefore, adapting long-tailed regression methods to video and motion prediction may be a promising direction, and one place to begin could be to apply the methods within Ren et al. (2022) and Yang et al. (2021) to motion and video prediction tasks.

6. Open Challenges

Throughout this paper, we have explored prediction algorithms that focus on long-tailed learning and social interaction modeling. Each of these areas can be considered fairly open challenges that are actively being worked on. However, in this section, we describe novel ways of thinking about these challenges, either by looking at them in combination with each other, or by combining them with other challenges in an insightful way.

6.1 Long-Tailed Open World Prediction

While datasets taken in open-ended environments represent many complex real-world interactions, it is still likely that systems like autonomous robots will encounter new classes of agents or unique agent behaviors unseen during training when they are released into the world. This is the open world problem: 1) improving performance on the long tail, and 2) recognizing when a new behavior or detection is one that hasn't been seen before (Liu et al., 2019). Dealing with the open world problem is a skill that autonomous robots operating in open-ended environments must acquire to perform effectively.

While the long-tailed open world framework has been recieving increasing attention within detection (e.g., Liu et al., 2019; Joseph et al., 2021; Saito et al., 2022; Konan et al., 2022), segmentation (e.g., Wang et al., 2021), and tracking (e.g., Liu et al., 2022; Dave, 2021), open-world prediction is still unexplored territory. One method that may be directly applicable to action prediction tasks is that proposed in Liu et al. (2019), which uses contrastive learning and a memory framework to improve performance on both common and uncommon classes, and recognize examples that don't fall into any of the given classes. Studying long-tailed open-world prediction may be a promising research direction, and one place to begin may be to apply the methods proposed in Liu et al. (2019) to action prediction.

6.2 Long-Tailed Environmental Factors

Most long-tailed learning algorithms focus on dataset imbalance across the distribution of labels or performances. Classification techniques separate examples based on how common the class label is, while some trajectory prediction algorithms separate trajectories by how different their error is from that of other examples (e.g., Makansi et al., 2021; Kozerawski et al., 2022). However, one underexplored aspect of long-tailedness is the influence of environmental factors. For example, in a dataset where most examples come from city roads, underrepresented country roads may lead to poor performance in the countryside. Or, if most examples are collected on sunny days, the system will be biased against rainy day samples. Current solutions include collecting more data from underrepresented environments, training multiple systems that reason differently in different environments (for example, one system for highway driving and one for urban driving), and domain adaptation, where knowledge is learned from a source domain like a sunny dataset, and transferred to a target domain, like a rainy dataset. However, all of these methods require environments to be separated into classes, which is unwieldy when environmental descriptors are regressive. For example, using time of day instead of classes like day and night, requires new methods that don't depend on the existence of environmental classes.

Therefore, further study on removing bias due to data imbalance across environmental factors is a necessary step towards helping networks perform in all environments. While long-tailed classification techniques like ensemble learning via multiple systems, transfer learning via domain adaptation, and domain balancing methods which adjust the loss based on self-supervised environment characteristics show some success, attempting to apply other methods such as those developed for long-tailed regression, may be a promising direction of research.

One area in particular where such an approach can be used is within trajectory prediction, where time of day and GPS location can be used as regressive environmental factors across which a dataset can be imbalanced. Consequently, if there is a particularly confusing intersection where accidents commonly happen, or a particular time of day, such as rush hour, where people behave differently, such unique environmental factors can be learned and modeled.

6.3 Long-Tailed Social Behavior

Another way in which behavior can be imbalanced is the extent to which agents follow social conventions or exhibit the social kinesics that let others know their intention. While most people adhere to these conventions and move in a way that indicates their intention, there are types of people and situations people may find themselves in where they deviate from these social norms. For example, if someone is rushing, they may prioritize speed over safety, quickly change lanes without giving signal, cut off another vehicle, or drive into the opposing lane in order to pass traffic. Children, on the other hand, may be to young to know about social conventions and may erratically change their mind about their intended destination, rendering their kinesics unusable.

While this type of behavior is difficult to predict, human drivers still anticipate and work around such behavior by identifying an unconventional agent, being aware of potential unexpected movements, and keeping a safe distance. Many previous works have studied anomaly detection within social interaction, where socially surprising behavior is catalogued as unique and risky (e.g., Chaker et al., 2017), but incorporating such anti-social risk assessment into prediction methods is just starting to be done like in Zhang et al. (2022), Jha et al. (2021). Zhang et al. (2022) attempts to identify risky vehicles on the road, while Jha et al. (2021) identifies risky situations as well as risky agents of all types. However, the risk prediction elements of these methods do not inform the path prediction elements, and are more of an 'uncertainty score' of the predicted path. Therefore, more work must be done in incorporating the risk score into the prediction modules so that the path can be predicted differently for unconventional agents.

Furthermore, these methods attempt to solve long-tailed social behavior from an anomaly detection perspective, where the goal is to detect a risky agent. While this is useful, it may also be beneficial to view this problem from a long-tailed learning perspective, through which uncommon behaviors can not only be differentiated, but also emphasized in the loss such that patterns within the domain of uncommon behaviors can be identified and used to predict the future.

7. Conclusion

In this work, we categorize the types of prediction tasks into a taxonomy and present a broad overview of the input encodings, modeling techniques, and training techniques of a selection of methods from each type of task. Furthermore, we review long-tailed learning within classification and regression, and highlight the major advances made in two relatively new areas of prediction that are being explored: social interaction modeling and long-tailed learning. Existing surveys have yet to focus on social interaction within open ended environments, or long-tailed regression and long-tailed prediction. Finally, we highlight upcoming research areas in need of further exploration.

References

AIcrowd — Trajnet++ (A Trajectory Forecasting Challenge) — Challenges. https://www.aicrowd.com/challenges/trajnet-a-trajectory-forecasting-challenge.

Figure 1. Pose estimation examples with our UniPose method.. https://www.researchgate.net/figure/Pose-estimation-examples-with-our-UniPose-method_fig1_338762733.

The history of autonomous vehicle datasets and 3 open-source Python apps for visualizing them — Modern Data. https://moderndata.plotly.com/the-history-of-autonomous-vehicle-datasets-and-3-open-source-python-apps-for-visualizing-them/.

Pedestrian trajectory prediction via the Social-Grid LSTM model - Cheng - 2018 - The Journal of Engineering - Wiley Online Library. https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/joe.2018.8316.

(2018). UC Berkeley open-sources self-driving dataset BDD100K. https://www.therobotreport.com/uc-berkeley-opens-self-driving-dataset-bdd100k/.

Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, P., Toderici, G., Varadarajan, B., & Vijayanarasimhan, S. (2016). YouTube-8M: A Large-Scale Video Classification Benchmark. *arXiv:1609.08675 [cs]*.

Adeli, V., Adeli, E., Reid, I., Niebles, J. C., & Rezatofighi, H. (2020). Socially and Contextually Aware Human Motion and Pose Forecasting. *IEEE Robotics and Automation Letters*, *5*(4), 6033–6040.

Adeli, V., Ehsanpour, M., Reid, I., Niebles, J. C., Savarese, S., Adeli, E., & Rezatofighi, H. (2021). TRiPOD: Human Trajectory and Pose Dynamics Forecasting in the Wild. *arXiv:2104.04029 [cs]*.

Airale, L., Vaufreydaz, D., & Alameda-Pineda, X. (2021). SocialInteractionGAN: Multi-person Interaction Sequence Generation. *arXiv:2103.05916 [cs, stat]*.

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., & Savarese, S. (2016). Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 961–971, Las Vegas, NV, USA. IEEE.

Anderson, C., Du, X., Vasudevan, R., & Johnson-Roberson, M. (2019). Stochastic Sampling Simulation for Pedestrian Trajectory Prediction. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4236–4243.

Andriluka, M., Pishchulin, L., Gehler, P., & Schiele, B. (2014). 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3686–3693, Columbus, OH, USA. IEEE.

Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R. H., & Levine, S. (2018). Stochastic Variational Video Prediction. *arXiv:1710.11252 [cs]*.

Balasubramaniam, A., & Pasricha, S. (2022). Object Detection in Autonomous Vehicles: Status and Open Challenges..

Barquero, G., Núñez, J., Escalera, S., Xu, Z., Tu, W.-W., Guyon, I., & Palmero, C. (2022). Didn't see that coming: A survey on non-verbal social human behavior forecasting. *arXiv:2203.02480 [cs]*.

Bhattacharjee, P., & Das, S. (2019). Predicting Video Frames Using Feature Based Locally Guided Objectives. In Jawahar, C., Li, H., Mori, G., & Schindler, K. (Eds.), *Computer Vision – ACCV 2018*, Vol. 11364, pp. 679–695. Springer International Publishing, Cham.

Bhattacharyya, A., Fritz, M., & Schiele, B. (2019). Bayesian Prediction of Future Street Scenes using Synthetic Likelihoods. *arXiv:1810.00746 [cs]*.

Bhattacharyya, A., Reino, D. O., Fritz, M., & Schiele, B. (2021). Euro-PVI: Pedestrian Vehicle Interactions in Dense Urban Centers. *arXiv:2106.12442 [cs]*.

Bock, J., Krajewski, R., Moers, T., Runde, S., Vater, L., & Eckstein, L. (2019). The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections..

Bock, J., Krajewski, R., Moers, T., Runde, S., Vater, L., & Eckstein, L. (2020). The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1929–1934.

Branco, P., Torgo, L., & Ribeiro, R. P. (2016). A Survey of Predictive Modeling on Imbalanced Domains. *ACM Computing Surveys*, *49*(2), 1–50.

Branco, P., Torgo, L., & Ribeiro, R. P. (2017). SMOGN: A Pre-processing Approach for Imbalanced Regression. In *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pp. 36–50. PMLR.

Byeon, W., Wang, Q., Srivastava, R. K., & Koumoutsakos, P. (2018). ContextVP: Fully Context-Aware Video Prediction. *arXiv:1710.08518 [cs]*.

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., & Beijbom, O. (2020). nuScenes: A multimodal dataset for autonomous driving. *arXiv:1903.11027 [cs, stat]*.

Cai, H., Bai, C., Tai, Y.-W., & Tang, C.-K. (2018). Deep Video Generation, Prediction and Completion of Human Action Sequences. In Ferrari, V., Hebert, M., Sminchisescu, C., & Weiss, Y. (Eds.), *Computer Vision – ECCV 2018*, Vol. 11206, pp. 374–390. Springer International Publishing, Cham.

Cao, W., Li, S., & Zhong, J. (2022). A dual attention model based on probabilistically mask for 3D human motion prediction. *Neurocomputing*, *493*, 106–118.

Chaker, R., Aghbari, Z. A., & Junejo, I. N. (2017). Social network model for crowd anomaly detection and localization. *Pattern Recognition*, *61*, 266–281.

Chan, A., Zeng, K., Mohapatra, P., Lee, S.-J., & Banerjee, S. (2010). Metrics for Evaluating Video Streaming Quality in Lossy IEEE 802.11 Wireless Networks. In *2010 Proceedings IEEE INFOCOM*, pp. 1–9, San Diego, CA, USA. IEEE.

Chandra, R., Bhattacharya, U., Bera, A., & Manocha, D. (2019). TraPHic: Trajectory Prediction in Dense and Heterogeneous Traffic Using Weighted Interactions. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8475–8484, Long Beach, CA, USA. IEEE.

Chandra, R., Guan, T., Panuganti, S., Mittal, T., Bhattacharya, U., Bera, A., & Manocha, D. (2020). Forecasting Trajectory and Behavior of Road-Agents Using Spectral Clustering in Graph-LSTMs. *arXiv:1912.01118 [cs]*.

Chang, M.-F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., & Hays, J. (2019). Argoverse: 3D Tracking and Forecasting with Rich Maps. *arXiv:1911.02620 [cs]*.

Chao, Y.-W., Yang, J., Price, B., Cohen, S., & Deng, J. (2017). Forecasting Human Dynamics from Static Images. *arXiv:1704.03432 [cs]*.

Chen, K., Song, X., & Ren, X. (2021). Pedestrian Trajectory Prediction in Heterogeneous Traffic Using Pose Keypoints-Based Convolutional Encoder-Decoder Network. *IEEE Transactions on Circuits and Systems for Video Technology*, *31*(5), 1764–1775.

Chen, L., Lu, J., Song, Z., & Zhou, J. (2022). Ambiguousness-Aware State Evolution for Action Prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 1–1.

Chen, T., Tian, R., & Ding, Z. (2021). Visual Reasoning using Graph Convolutional Networks for Predicting Pedestrian Crossing Intention. In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pp. 3096–3102, Montreal, BC, Canada. IEEE.

Cheriyadat, A., & Radke, R. (2008). Detecting Dominant Motions in Dense Crowds. *IEEE Journal of Selected Topics in Signal Processing*, *2*(4), 568–581.

Cho, J., Lee, J., Oh, C., Song, W., & Sohn, K. (2021). Wide and Narrow: Video Prediction from Context and Motion. *arXiv:2110.11586 [cs]*.

Cho, S., & Foroosh, H. (2018). A Temporal Sequence Learning for Action Recognition and Prediction. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 352–361.

Choi, C., & Dariush, B. (2019). Looking to Relations for Future Trajectory Forecast. *arXiv:1905.08855 [cs]*.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., & Schiele, B. (2016). The Cityscapes Dataset for Semantic Urban Scene Understanding. *arXiv:1604.01685 [cs]*.

Corona, E., Pumarola, A., Alenyà, G., & Moreno-Noguer, F. (2020). Context-aware Human Motion Prediction. *arXiv:1904.03419 [cs]*.

Dave, A. (2021). *Open-World Object Detection and Tracking*. Ph.D. thesis, Carnegie Mellon University.

De Geest, R., Gavves, E., Ghodrati, A., Li, Z., Snoek, C., & Tuytelaars, T. (2016). Online Action Detection. In Leibe, B., Matas, J., Sebe, N., & Welling, M. (Eds.), *Com-*

puter Vision – ECCV 2016, Lecture Notes in Computer Science, pp. 269–284, Cham. Springer International Publishing.

Devarakonda, H., & Mukherjee, S. (2021). Early Prediction of Human Action by Deep Reinforcement Learning. In *2021 National Conference on Communications (NCC)*, pp. 1–6.

Dollar, P., Wojek, C., Schiele, B., & Perona, P. (2009). Pedestrian detection: A benchmark. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 304–311.

Dong, B., Liu, H., Bai, Y., Lin, J., Xu, Z., Xu, X., & Kong, Q. (2021). Multi-modal Trajectory Prediction for Autonomous Driving with Semantic Map and Dynamic Graph Attention Network. *arXiv:2103.16273 [cs]*.

Dwivedi, I., Malla, S., Dariush, B., & Choi, C. (2020). SSP: Single Shot Future Trajectory Prediction. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2211–2218.

Fan, L., Buch, S., Wang, G., Cao, R., Zhu, Y., Niebles, J. C., & Fei-Fei, L. (2020). RubiksNet: Learnable 3D-Shift for Efficient Video Action Recognition. In Vedaldi, A., Bischof, H., Brox, T., & Frahm, J.-M. (Eds.), *Computer Vision – ECCV 2020*, Vol. 12364, pp. 505–521. Springer International Publishing, Cham.

Feng, W., Kannan, A., Gkioxari, G., & Zitnick, C. L. (2017). Learn2Smile: Learning non-verbal interaction through observation. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4131–4138.

Fragkiadaki, K., Huang, J., Alemi, A., Vijayanarasimhan, S., Ricco, S., & Sukthankar, R. (2017). Motion Prediction Under Multimodality with Conditional Stochastic Networks. *arXiv:1705.02082 [cs]*.

Frossard, D., Kee, E., & Urtasun, R. (2019). DeepSignals: Predicting Intent of Drivers Through Visual Signals. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9697–9703.

Fugošić, K., Šarić, J., & Šegvić, S. (2020). Multimodal semantic forecasting based on conditional generation of future features. *arXiv:2010.09067 [cs]*.

Gammulle, H., Denman, S., Sridharan, S., & Fookes, C. (2019). Predicting the Future: A Jointly Learnt Model for Action Anticipation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5562–5571.

Geiger, A., Lenz, P., & Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361.

Georgiou, H., Karagiorgou, S., Kontoulis, Y., Pelekis, N., Petrou, P., Scarlatti, D., & Theodoridis, Y. (2018). Moving Objects Analytics: Survey on Future Location & Trajectory Prediction Methods. *arXiv:1807.04639 [cs, stat]*.

Gesnouin, J., Pechberti, S., Stanciulcscu, B., & Moutarde, F. (2021). TrouSPI-Net: Spatio-temporal attention on parallel atrous convolutions and U-GRUs for skeletal pedestrian crossing prediction. In *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021)*, pp. 01–07.

Girase, H., Gang, H., Malla, S., Li, J., Kanehara, A., Mangalam, K., & Choi, C. (2021). LOKI: Long Term and Key Intentions for Trajectory Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9803–9812.

Gopalakrishnan, A., Mali, A., Kifer, D., Giles, C. L., & Ororbia, A. G. (2019). A Neural Temporal Model for Human Motion Prediction. *arXiv:1809.03036 [cs]*.

Gu, C., Zhao, Y., & Zhang, C. (2021). Learning to Predict Diverse Human Motions from a Single Image via Mixture Density Networks. *arXiv:2109.05776 [cs]*.

Gu, T., Chen, G., Li, J., Lin, C., Rao, Y., Zhou, J., & Lu, J. (2022). Stochastic Trajectory Prediction via Motion Indeterminacy Diffusion. *arXiv:2203.13777 [cs]*.

Gui, L.-Y., Zhang, K., Wang, Y.-X., Liang, X., Moura, J. M. F., & Veloso, M. (2018). Teaching Robots to Predict Human Motion. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 562–567, Madrid. IEEE.

Gulzar, M., Muhammad, Y., & Muhammad, N. (2021). A Survey on Motion Prediction of Pedestrians and Vehicles for Autonomous Driving. *IEEE Access*, *9*, 137957–137969.

Gundavarapu, A., Chakravarthy, V. S., & Soman, K. (2019). A Model of Motion Processing in the Visual Cortex Using Neural Field With Asymmetric Hebbian Learning. *Frontiers in Neuroscience*, *13*.

Haddad, S., & Lam, S.-K. (2021). Self-Growing Spatial Graph Network for Context-Aware Pedestrian Trajectory Prediction. In *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 1029–1033.

Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., & Bing, G. (2017). Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, *73*, 220–239.

Hara, K., Kataoka, H., & Satoh, Y. (2018). Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?..

Hasan, I., Setti, F., Tsesmelis, T., Belagiannis, V., Amin, S., Del Bue, A., Cristani, M., & Galasso, F. (2021). Forecasting People Trajectories and Head Poses by Jointly Reasoning on Tracklets and Vislets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(4), 1267–1278.

Hasan, I., Setti, F., Tsesmelis, T., Del Bue, A., Cristani, M., & Galasso, F. (2018). "Seeing is Believing": Pedestrian Trajectory Forecasting Using Visual Frustum of Attention. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1178–1185, Lake Tahoe, NV. IEEE.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition..

Hirakawa, T., Yamashita, T., Tamaki, T., & Fujiyoshi, H. (2018). Survey on Vision-based Path Prediction. *arXiv:1811.00233 [cs]*, *10922*, 48–64.

Ho, Y.-H., Cho, C.-Y., Jin, G.-L., & Peng, W.-H. (2019). SME-Net: Sparse Motion Estimation for Parametric Video Prediction Through Reinforcement Learning. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10461–10469, Seoul, Korea (South). IEEE.

Houston, J., Zuidhof, G., Bergamini, L., Ye, Y., Chen, L., Jain, A., Omari, S., Iglovikov, V., & Ondruska, P. (2020). One Thousand and One Hours: Self-driving Motion Prediction Dataset. *arXiv:2006.14480 [cs]*.

Hu, X., Dai, J., Li, M., Peng, C., Li, Y., & Du, S. (2022). Online human action detection and anticipation in videos: A survey. *Neurocomputing*, *491*, 395–413.

Huang, Y., Bi, H., Li, Z., Mao, T., & Wang, Z. (2019). STGAT: Modeling Spatial-Temporal Interactions for Human Trajectory Prediction. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6271–6280, Seoul, Korea (South). IEEE.

Huang, Y., & Khan, S. M. (2017). DyadGAN: Generating Facial Expressions in Dyadic Interactions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 11–18.

Hung, H., & Krose, B. (2011). Detecting F-formations as dominant sets. In *Journal of Theoretical Biology - J THEOR BIOL*, pp. 231–238.

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift..

Ionescu, C., Papava, D., Olaru, V., & Sminchisescu, C. (2014). Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(7), 1325–1339.

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2018). Image-to-Image Translation with Conditional Adversarial Networks..

Jadon, S. (2020). A survey of loss functions for semantic segmentation. In *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pp. 1–7.

Jain, A., Zamir, A. R., Savarese, S., & Saxena, A. (2016). Structural-RNN: Deep Learning on Spatio-Temporal Graphs. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5308–5317, Las Vegas, NV, USA. IEEE.

Jha, R. B., Rai, A., & Kala, R. (2021). Predictive Risk Analysis using Deep Learning in Indian Traffic. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 258–263.

Jhuang, H., Gall, J., Zuffi, S., Schmid, C., & Black, M. J. (2013). Towards Understanding Action Recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3192–3199.

Johnson, J. M., & Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*, *6*(1), 27.

Joo, H., Simon, T., Cikara, M., & Sheikh, Y. (2019). Towards Social Artificial Intelligence: Nonverbal Social Signal Prediction in a Triadic Interaction. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10865–10875, Long Beach, CA, USA. IEEE.

Joseph, K. J., Khan, S., Khan, F. S., & Balasubramanian, V. N. (2021). Towards Open World Object Detection. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5826–5836, Nashville, TN, USA. IEEE.

Kanazawa, A., Zhang, J. Y., Felsen, P., & Malik, J. (2019). Learning 3D Human Dynamics From Video. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5607–5616, Long Beach, CA, USA. IEEE.

Kao, I.-H., & Chan, C.-Y. (2022). Impact of Posture and Social Features on Pedestrian Road-Crossing Trajectory Prediction. *IEEE Transactions on Instrumentation and Measurement*, *71*, 1–16.

Kao, I.-H., Zhou, X., Chen, I.-M., Wang, P., & Chan, C.-Y. (2021). A Posture Features Based Pedestrian Trajectory Prediction with LSTM. In *2021 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, pp. 1–2.

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M., & Zisserman, A. (2017). The Kinetics Human Action Video Dataset..

Konan, S., Liang, K. J., & Yin, L. (2022). Extending One-Stage Detection with Open-World Proposals. *arXiv:2201.02302 [cs]*.

Kong, Y., Jia, Y., & Fu, Y. (2012). Learning Human Interaction by Interactive Phrases. In Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., & Schmid, C. (Eds.), *Computer Vision – ECCV 2012*, pp. 300–313, Berlin, Heidelberg. Springer.

Korbmacher, R., & Tordeux, A. (2021). Review of Pedestrian Trajectory Prediction Methods: Comparing Deep Learning and Knowledge-based Approaches. *arXiv:2111.06740 [physics, stat]*.

Kosaraju, V., Sadeghian, A., Martín-Martín, R., Reid, I., Rezatofighi, S. H., & Savarese, S. (2019). Social-BiGAT: Multimodal Trajectory Forecasting using Bicycle-GAN and Graph Attention Networks. *arXiv:1907.03395 [cs]*.

Kothari, P., Kreiss, S., & Alahi, A. (2021). Human Trajectory Forecasting in Crowds: A Deep Learning Perspective..

Kotseruba, I., Rasouli, A., & Tsotsos, J. K. (2020). Joint Attention in Autonomous Driving (JAAD). *arXiv:1609.04741 [cs]*.

Kozerawski, J., Sharan, M., & Yu, R. (2022). Taming the Long Tail of Deep Probabilistic Forecasting. *arXiv:2202.13418 [cs]*.

Krajewski, R., Moers, T., Bock, J., Vater, L., & 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)*, pp. 1–6.

Krawczyk, B. (2016). Learning from imbalanced data: Open challenges and future directions. *Progress in Artificial Intelligence*, *5*(4), 221–232.

Kwon, Y.-H., & Park, M.-G. (2019). Predicting Future Frames Using Retrospective Cycle GAN. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1811–1820, Long Beach, CA, USA. IEEE.

Lee, K.-H., Tagawa, T., Pan, J.-E. M., Gaidon, A., & Douillard, B. (2019). An Attention-based Recurrent Convolutional Network for Vehicle Taillight Recognition. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2365–2370.

Leon, F., & Gavrilescu, M. (2021). A Review of Tracking and Trajectory Prediction Methods for Autonomous Driving. *Mathematics*, *9*(6), 660.

Lerner, A., Chrysanthou, Y., & Lischinski, D. (2007). Crowds by Example. *Comput. Graph. Forum*.

Li, B., Tian, J., Zhang, Z., Feng, H., & Li, X. (2021a). Multitask Non-Autoregressive Model for Human Motion Prediction. *IEEE Transactions on Image Processing*, *30*, 2562–2574.

Li, X., Rosman, G., Gilitschenski, I., DeCastro, J., Vasile, C.-I., Karaman, S., & Rus, D. (2021b). Differentiable Logic Layer for Rule Guided Trajectory Prediction. In *Proceedings of the 2020 Conference on Robot Learning*, pp. 2178–2194. PMLR.

Liang, X., Lee, L., Dai, W., & Xing, E. P. (2017). Dual Motion GAN for Future-Flow Embedded Video Prediction. *arXiv:1708.00284 [cs]*.

Lin, J., Gan, C., & Han, S. (2019). TSM: Temporal Shift Module for Efficient Video Understanding..

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2018). Focal Loss for Dense Object Detection..

Liu, B., Adeli, E., Cao, Z., Lee, K.-H., Shenoi, A., Gaidon, A., & Niebles, J. C. (2020). Spatiotemporal Relationship Reasoning for Pedestrian Intent Prediction. *arXiv:2002.08945 [cs]*.

Liu, J., Mao, X., Fang, Y., Zhu, D., & Meng, M. Q.-H. (2021). A Survey on Deep-Learning Approaches for Vehicle Trajectory Prediction in Autonomous Driving. *arXiv:2110.10436 [cs]*.

Liu, W., Sharma, A., Camps, O., & Sznaier, M. (2018). DYAN: A Dynamical Atoms-Based Network for Video Prediction. In Ferrari, V., Hebert, M., Sminchisescu, C., & Weiss, Y. (Eds.), *Computer Vision – ECCV 2018*, Vol. 11216, pp. 175–191. Springer International Publishing, Cham.

Liu, X., Liu, X., & Yin, J. (2021). A Discussion of Data Sampling Strategies for Early Action Prediction..

Liu, Y., Zulfikar, I. E., Luiten, J., Dave, A., Ramanan, D., Leibe, B., Ošep, A., & Leal-Taixé, L. (2022). Opening up Open-World Tracking. *arXiv:2104.11221 [cs]*.

Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., & Yu, S. X. (2019). Large-Scale Long-Tailed Recognition in an Open World. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2532–2541, Long Beach, CA, USA. IEEE.

Lorenzo, J., Parra, I., & Sotelo, M. A. (2021a). IntFormer: Predicting pedestrian intention with the aid of the Transformer architecture. *arXiv:2105.08647 [cs]*.

Lorenzo, J., Alonso, I. P., Izquierdo, R., Ballardini, A. L., Saz, A. H., Llorca, D. F., & Sotelo, M. A. (2021b). CAPformer: Pedestrian Crossing Action Prediction Using Transformer. *Sensors*, *21*(17), 5694.

Luc, P., Neverova, N., Couprie, C., Verbeek, J., & LeCun, Y. (2017). Predicting Deeper into the Future of Semantic Segmentation..

Luo, W., Liu, W., & Gao, S. (2017). A Revisit of Sparse Coding Based Anomaly Detection in Stacked RNN Framework. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 341–349, Venice. IEEE.

Ma, Y., ZHU, X., Cheng, X., Yang, R., Liu, J., & Manocha, D. (2020). AutoTrajectory: Label-free Trajectory Extraction and Prediction from Videos using Dynamic Points..

Makansi, O., Cicek, O., Marrakchi, Y., & Brox, T. (2021). On Exposing the Challenging Long Tail in Future Prediction of Traffic Actors. *arXiv:2103.12474 [cs]*.

Makansi, O., Ilg, E., Cicek, O., & Brox, T. (2019). Overcoming Limitations of Mixture Density Networks: A Sampling and Fitting Framework for Multimodal Future Prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7144–7153.

Malla, S., Dariush, B., & Choi, C. (2020). TITAN: Future Forecast Using Action Priors. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11183–11193, Seattle, WA, USA. IEEE.

Mangalam, K., An, Y., Girase, H., & Malik, J. (2020a). From Goals, Waypoints & Paths To Long Term Human Trajectory Forecasting. *arXiv:2012.01526 [cs]*.

Mangalam, K., Girase, H., Agarwal, S., Lee, K.-H., Adeli, E., Malik, J., & Gaidon, A. (2020b). It Is Not the Journey but the Destination: Endpoint Conditioned Trajectory Prediction. *arXiv:2004.02025 [cs]*.

Mao, W., Liu, M., & Salzmann, M. (2020). History Repeats Itself: Human Motion Prediction via Motion Attention. *arXiv:2007.11755 [cs, eess]*.

Mao, W., Liu, M., Salzmann, M., & Li, H. (2019). Learning Trajectory Dependencies for Human Motion Prediction. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9488–9496, Seoul, Korea (South). IEEE.

Min, W., Ha, E. Y., Rowe, J., Mott, B., & Lester, J. (2014). Deep Learning-Based Goal Recognition in Open-Ended Digital Games. In *Tenth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning..

Moniz, N., Branco, P., & Torgo, L. (2017). Evaluation of Ensemble Methods in Imbalanced Regression Tasks. In *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pp. 129–140. PMLR.

Nasrabadi, A. T., Shirsavar, M. A., Ebrahimi, A., & Ghanbari, M. (2014). Investigating the PSNR calculation methods for video sequences with source and channel distortions. In *2014 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, pp. 1–4.

Oksuz, K., Cam, B. C., Kalkan, S., & Akbas, E. (2021). Imbalance Problems in Object Detection: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(10), 3388–3415.

Oliu, M., Selva, J., & Escalera, S. (2018). Folded Recurrent Neural Networks for Future Video Prediction. *arXiv:1712.00311 [cs, stat]*.

Oprea, S., Martinez-Gonzalez, P., Garcia-Garcia, A., Castro-Vargas, J. A., Orts-Escolano, S., Garcia-Rodriguez, J., & Argyros, A. (2022). A Review on Deep Learning Techniques for Video Prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(6), 2806–2826.

Özyer, T., Ak, D. S., & Alhajj, R. (2021). Human action recognition approaches with video datasets—A survey. *Knowledge-Based Systems*, *222*, 106995.

Paravarzar, S., & Mohammad, B. (2020). Motion Prediction on Self-driving Cars: A Review. *arXiv:2011.03635 [cs]*.

Patron-Perez, A., Marszalek, M., Zisserman, A., & Reid, I. (2010). High Five: Recognising human interactions in TV shows.. In *BMVC*, Vol. 1, p. 33. Citeseer.

Pellegrini, S., Ess, A., & Van Gool, L. (2010). Improving Data Association by Joint Modeling of Pedestrian Trajectories and Groupings. In Daniilidis, K., Maragos, P., & Paragios, N. (Eds.), *Computer Vision – ECCV 2010*, pp. 452–465, Berlin, Heidelberg. Springer.

Perez, M., Liu, J., & Kot, A. C. (2021). Interaction Relational Network for Mutual Action Recognition. *arXiv:1910.04963 [cs]*.

Rasouli, A. (2020). Deep Learning for Vision-based Prediction: A Survey. *arXiv:2007.00095 [cs]*.

Rasouli, A., Kotseruba, I., Kunic, T., & Tsotsos, J. (2019). PIE: A Large-Scale Dataset and Models for Pedestrian Intention Estimation and Trajectory Prediction. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6261–6270, Seoul, Korea (South). IEEE.

Rasouli, A., Kotseruba, I., & Tsotsos, J. K. (2017). Are They Going to Cross? A Benchmark Dataset and Baseline for Pedestrian Crosswalk Behavior. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 206–213, Venice, Italy. IEEE.

Rasouli, A., Rohani, M., & Luo, J. (2021). Bifold and Semantic Reasoning for Pedestrian Behavior Prediction. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15580–15590, Montreal, QC, Canada. IEEE.

Rasouli, A., Yau, T., Lakner, P., Malekmohammadi, S., Rohani, M., & Luo, J. (2020a). PePScenes: A Novel Dataset and Baseline for Pedestrian Action Prediction in 3D. *arXiv:2012.07773 [cs]*.

Rasouli, A., Yau, T., Rohani, M., & Luo, J. (2020b). Multi-Modal Hybrid Architecture for Pedestrian Action Prediction. *arXiv:2012.00514 [cs]*.

Ren, J., Zhang, M., Yu, C., & Liu, Z. (2022). Balanced MSE for Imbalanced Visual Regression. *arXiv:2203.16427 [cs]*.

Richter, C., Barragán, P. R., & Karaman, S. (2022). Learning and Predicting Multimodal Vehicle Action Distributions in a Unified Probabilistic Model Without Labels..

Robicquet, A., Sadeghian, A., Alahi, A., & Savarese, S. (2016). Learning Social Etiquette: Human Trajectory Understanding In Crowded Scenes. In Leibe, B., Matas, J., Sebe, N., & Welling, M. (Eds.), *Computer Vision – ECCV 2016*, Vol. 9912, pp. 549–565. Springer International Publishing, Cham.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In Navab, N., Hornegger, J., Wells, W. M., & Frangi, A. F. (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Vol. 9351, pp. 234–241. Springer International Publishing, Cham.

Rosenfeld, A., & Ullman, S. (2018). Action Classification via Concepts and Attributes..

Rudenko, A., Palmieri, L., Herman, M., Kitani, K. M., Gavrila, D. M., & Arras, K. O. (2020). Human Motion Trajectory Prediction: A Survey. *The International Journal of Robotics Research*, *39*(8), 895–935.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge..

Ryoo, M. S., & Aggarwal, J. K. (2010). UT-interaction dataset, ICPR contest on semantic description of human activities (SDHA). In *IEEE International Conference on Pattern Recognition Workshops*, Vol. 2, p. 4.

Sadeghian, A., Kosaraju, V., Gupta, A., Savarese, S., & Alahi, A. (2018a). Trajnet: Towards a benchmark for human trajectory prediction. *arXiv preprint*.

Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Rezatofighi, S. H., & Savarese, S. (2018b). SoPhie: An Attentive GAN for Predicting Paths Compliant to Social and Physical Constraints. *arXiv:1806.01482 [cs]*.

Saito, K., Hu, P., Darrell, T., & Saenko, K. (2022). Learning to Detect Every Thing in an Open World. *arXiv:2112.01698 [cs]*.

Salzmann, T., Ivanovic, B., Chakravarty, P., & Pavone, M. (2021). Trajectron++: Dynamically-Feasible Trajectory Forecasting With Heterogeneous Data. *arXiv:2001.03093 [cs]*.

Sampath, V., Maurtua, I., Aguilar Martín, J. J., & Gutierrez, A. (2021). A survey on generative adversarial networks for imbalance problems in computer vision tasks. *Journal of Big Data*, *8*(1), 27.

Sanghvi, N., Yonetani, R., & Kitani, K. (2020). MGpi: A Computational Model of Multiagent Group Perception and Interaction. *arXiv:1903.01537 [cs, stat]*.

Schuldt, C., Laptev, I., & Caputo, B. (2004). Recognizing human actions: A local SVM approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, Vol. 3, pp. 32–36 Vol.3.

Shi, Y., Fernando, B., & Hartley, R. (2018). Action Anticipation with RBF Kernelized Feature Mapping RNN. In Ferrari, V., Hebert, M., Sminchisescu, C., & Weiss, Y. (Eds.), *Computer Vision – ECCV 2018*, Vol. 11214, pp. 305–322. Springer International Publishing, Cham.

Soomro, K., Zamir, A. R., & Shah, M. (2012). UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. *arXiv:1212.0402 [cs]*.

Steininger, M., Kobs, K., Davidson, P., Krause, A., & Hotho, A. (2021). Density-based weighting for imbalanced regression. *Machine Learning*, *110*(8), 2187–2211.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015). Rethinking the Inception Architecture for Computer Vision..

Tamaru, R., Siritanawan, P., & Kotani, K. (2021). Interaction Aware Relational Representations for Video Prediction. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2089–2094.

Tao, Z., Bai, Y., Zhao, H., Li, S., Kong, Y., & Fu, Y. (2021). Adversarial Memory Networks for Action Prediction. *arXiv:2112.09875 [cs]*.

Tran, V., Balasubramanian, N., & Hoai, M. (2021). Progressive Knowledge Distillation For Early Action Recognition. In *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 2583–2587.

van Doorn, F. (2018). Rituals of Leaving: Predictive Modelling of Leaving Behaviour in Conversation..

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need..

von Marcard, T., Henschel, R., Black, M. J., Rosenhahn, B., & Pons-Moll, G. (2018). Recovering Accurate 3D Human Pose in the Wild Using IMUs and a Moving Camera. In Ferrari, V., Hebert, M., Sminchisescu, C., & Weiss, Y. (Eds.), *Computer Vision – ECCV 2018*, Vol. 11214, pp. 614–631. Springer International Publishing, Cham.

Vrigkas, M., Nikou, C., & Kakadiaris, I. A. (2015). A Review of Human Activity Recognition Methods. *Frontiers in Robotics and AI*, *2*.

Wang, A., & Steinfeld, A. (2020). Group Split and Merge Prediction With 3D Convolutional Networks. *IEEE Robotics and Automation Letters*, *5*(2), 1923–1930.

Wang, B., Adeli, E., Chiu, H.-k., Huang, D.-A., & Niebles, J. C. (2019). Imitation Learning for Human Pose Prediction. *arXiv:1909.03449 [cs]*.

Wang, C., Wang, Y., Xu, M., & Crandall, D. J. (2022). Stepwise Goal-Driven Networks for Trajectory Prediction. *IEEE Robotics and Automation Letters*, *7*(2), 2716–2723.

Wang, J., Xu, H., Narasimhan, M., & Wang, X. (2021). Multi-Person 3D Motion Prediction with Multi-Range Transformers. *arXiv:2111.12073 [cs]*.

Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., & Van Gool, L. (2016). Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. In Leibe, B., Matas, J., Sebe, N., & Welling, M. (Eds.), *Computer Vision – ECCV 2016*, Vol. 9912, pp. 20–36. Springer International Publishing, Cham.

Wang, W., Feiszli, M., Wang, H., & Tran, D. (2021). Unidentified Video Objects: A Benchmark for Dense, Open-World Segmentation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10756–10765, Montreal, QC, Canada. IEEE.

Wang, X., Hu, J.-F., Lai, J.-H., Zhang, J., & Zheng, W.-S. (2019). Progressive Teacher-Student Learning for Early Action Prediction. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3551–3560, Long Beach, CA, USA. IEEE.

Wang, Z., Bovik, A., Sheikh, H., & Simoncelli, E. (2004). Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, *13*(4), 600–612.

Wiles, O., Koepke, A. S., & Zisserman, A. (2018). Self-supervised learning of a facial attribute embedding from video. *arXiv:1808.06882 [cs]*.

Wong, C., Xia, B., Hong, Z., Peng, Q., & You, X. (2021). View Vertically: A Hierarchical Network for Trajectory Prediction via Fourier Spectrums. *arXiv:2110.07288 [cs]*.

Wu, X., Zhao, J., & Wang, R. (2021). Anticipating Future Relations via Graph Growing for Action Prediction. In *AAAI*.

Xu, P., Hayet, J.-B., & Karamouzas, I. (2022). SocialVAE: Human Trajectory Prediction using Timewise Latents. *arXiv:2203.08207 [cs]*.

Xu, Y., Yang, X., Gong, L., Lin, H.-C., Wu, T.-Y., Li, Y., & Vasconcelos, N. (2020). Explainable Object-Induced Action Decision for Autonomous Vehicles. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9520–9529, Seattle, WA, USA. IEEE.

Yadav, S. K., Tiwari, K., Pandey, H. M., & Akbar, S. A. (2021). A review of multimodal human activity recognition with special emphasis on classification, applications, challenges and future directions. *Knowledge-Based Systems*, *223*, 106970.

Yan, X., Rastogi, A., Villegas, R., Sunkavalli, K., Shechtman, E., Hadap, S., Yumer, E., & Lee, H. (2018). MT-VAE: Learning Motion Transformations to Generate Multimodal Human Dynamics. In Ferrari, V., Hebert, M., Sminchisescu, C., & Weiss, Y. (Eds.), *Computer Vision – ECCV 2018*, Vol. 11209, pp. 276–293. Springer International Publishing, Cham.

Yang, Y., Zha, K., Chen, Y., Wang, H., & Katabi, D. (2021). Delving into Deep Imbalanced Regression. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 11842–11851. PMLR.

Yao, Y., Atkins, E., Roberson, M. J., Vasudevan, R., & Du, X. (2021). Coupling Intent and Action for Pedestrian Crossing Behavior Prediction. *arXiv:2105.04133 [cs]*.

Yao, Y., Xu, M., Choi, C., Crandall, D. J., Atkins, E. M., & Dariush, B. (2019). Egocentric Vision-based Future Vehicle Localization for Intelligent Driving Assistance Systems. *arXiv:1809.07408 [cs]*.

Yasar, M. S., & Iqbal, T. (2021). A Scalable Approach to Predict Multi-Agent Motion for Human-Robot Collaboration. *IEEE Robotics and Automation Letters*, *6*(2), 1686–1693.

Ying, G., Zou, Y., Wan, L., Hu, Y., & Feng, J. (2019). *Better Guider Predicts Future Better: Difference Guided Generative Adversarial Networks.*

Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., & Darrell, T. (2020). BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2633–2642, Seattle, WA, USA. IEEE.

Yuan, Y., & Kitani, K. (2019). Ego-Pose Estimation and Forecasting as Real-Time PD Control. *arXiv:1906.03173 [cs]*.

Zaech, J.-N., Dai, D., Liniger, A., & Gool, L. V. (2020). Action Sequence Predictions of Vehicles in Urban Environments using Map and Social Context. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8982–8989.

Zang, C., Pei, M., & Kong, Y. (2020). Few-shot Human Motion Prediction via Learning Novel Motion Dynamics. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 846–852, Yokohama, Japan. International Joint Conferences on Artificial Intelligence Organization.

Zhan, W., Sun, L., Wang, D., Shi, H., Clausse, A., Naumann, M., Kummerle, J., Konigshof, H., Stiller, C., de La Fortelle, A., & Tomizuka, M. (2019). INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps. *arXiv:1910.03088 [cs, eess]*.

Zhang, C., Chen, T., Liu, H., Shen, Q., & Ma, Z. (2019). Looking-Ahead: Neural Future Video Frame Prediction. In *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 1975–1979, Taipei, Taiwan. IEEE.

Zhang, E., Masoud, N., Bandegi, M., & Malhan, R. K. (2022). Predicting Risky Driving in a Connected Vehicle Environment. *IEEE Transactions on Intelligent Transportation Systems*, 1–12.

Zhang, L., She, Q., & Guo, P. (2019). Stochastic trajectory prediction with social graph network. *arXiv:1907.10233 [cs]*.

Zhang, W., Zhu, M., & Derpanis, K. G. (2013). From Actemes to Action: A Strongly-Supervised Representation for Detailed Action Understanding. In *2013 IEEE International Conference on Computer Vision*, pp. 2248–2255, Sydney, Australia. IEEE.

Zhang, X., Fang, Z., Wen, Y., Li, Z., & Qiao, Y. (2016). Range Loss for Deep Face Recognition with Long-tail..

Zhang, X., Wu, Z., Weng, Z., Fu, H., Chen, J., Jiang, Y.-G., & Davis, L. S. (2021a). VideoLT: Large-Scale Long-Tailed Video Recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7960–7969.

Zhang, Y., Kang, B., Hooi, B., Yan, S., & Feng, J. (2021b). Deep Long-Tailed Learning: A Survey. *arXiv:2110.04596 [cs]*.

Zhang, Z., Luo, P., Loy, C. C., & Tang, X. (2017). From Facial Expression Recognition to Interpersonal Relation Prediction. *arXiv:1609.06426 [cs]*.

Zhao, H., & Wildes, R. P. (2021). Review of Video Predictive Understanding: Early Action Recognition and Future Action Prediction. *arXiv:2107.05140 [cs]*.

Zhao, T., Xu, Y., Monfort, M., Choi, W., Baker, C., Zhao, Y., Wang, Y., & Wu, Y. N. (2019). Multi-Agent Tensor Fusion for Contextual Trajectory Prediction. *arXiv:1904.04776 [cs]*.

Zhou, H., Ren, D., Yang, X., Fan, M., & Huang, H. (2021). Sliding Sequential CVAE with Time Variant Socially-aware Rethinking for Trajectory Prediction. *arXiv:2110.15016 [cs]*.

Zhou, Y., Dong, H., & El Saddik, A. (2020). Deep Learning in Next-Frame Prediction: A Benchmark Review. *IEEE Access*, *8*, 69273–69283.