Learning to Predict Human Behavior from Video

Panna Felsen

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2019-66

http://www2.eecs.berkeley.edu/Pubs/TechRpts/2019/EECS-2019-66.html

May 17, 2019

Learning to Predict Human Behavior from Video

by

Panna Felsen

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Jitendra Malik, Chair
Professor Alexei Efros
Professor Martin Banks

Spring 2019

Learning to Predict Human Behavior from Video

Abstract

Learning to Predict Human Behavior from Video

by

Panna Felsen

Doctor of Philosophy in Engineering - Electrical Engineering and Computer Science

University of California, Berkeley

Professor Jitendra Malik, Chair

In recent times, the field of computer vision has made great progress with recognizing and tracking people and their activities in videos. However, for systems designed to interact dynamically with humans, tracking and recognition are insufficient; the ability to predict behavior is requisite. In this thesis, we introduce various general frameworks for predict human behavior at three levels of granularity: events, motion, and dynamics. In Chapter 2, we present a system that is capable of predicting future events. In Chapter 3, we present a system that is capable of personalized prediction of the future motion of multi-agent, adversarial interactions. Finally, in Chapter 4, we present a framework for learning a representation of human dynamics that we can: 1) use to estimate the 3d pose and shape of people moving in videos, and 2) use to hallucinate the motion surrounding a single-frame snapshot. We conclude with several promising future directions for learning to predict human behavior from video.

to my parents, Patti and Attila

Contents

Contents ii

1 Introduction 1

2 Discrete Prediction: Predicting Events 4
2.1 Introduction 4
2.2 Related Work 6
2.3 Datasets 6
2.4 Methods: From Images to Overhead View 9
2.5 Experiments 10
2.6 Discussion 18

3 Continuous Prediction: Predicting 2D Motion 20
3.1 Introduction 20
3.2 Related Work 22
3.3 Basketball Tracking Dataset 24
3.4 Methods 24
3.5 Experiments 28
3.6 Discussion 34

4 Learning 3D Human Dynamics 35
4.1 Introduction 35
4.2 Related Work 37
4.3 Approach 38
4.4 Learning from Unlabeled Video 42
4.5 Experimental Setup 43
4.6 Experiments 44
4.7 Discussion 48
4.8 Supplementary 48

5 Conclusion 52

Bibliography 54

Acknowledgments

Thank you Jitendra Malik for letting me pursue projects that I was excited about, and for giving me time to explore many areas of computer vision before settling on a thesis direction. Thank you for sharing with me not only your philosophies, but also many articles throughout the years, on how to do research, what makes a problem worthwhile, how to write a paper, and the importance of (considering) figure 1.

Thank you Alyosha Efros and Martin Banks for serving on both my thesis and qualifying committees, and thank you Trevor Darrell for serving on my qualifying committee.

Thank you to all of my wonderful collaborators. To thank you for everything that I learned from each of you would fill this entire thesis and more. Angjoo Kanazawa, your energy and enthusiasm is positively infectious. Jason Zhang, not only has it been a joy to re-experience computer vision through fresh eyes, but you have a knack for implementing things both quickly and well. Pulkit Agrawal, your calm demeanor makes even the most stressful deadline feel doable. Patrick Lucey and Sujoy Ganguly, thank you for sharing with me your vision for building systems that understand sports, and your passion for predicting and analyzing sports matches and athletes' performance. Katerina Fragkiadaki, it's very simple: thank you for your guidance.

Thank you everyone at Berkeley, especially members of Jitendra's, Alyosha's, and Trevor's groups over the years: Andrew, Ashish, Bharath, Deepak, Georgia, Jasmine, Jon, Judy, Jun-Yan, Ke, Lisa Anne, Pablo, Philipp, Phillip, Richard, Ross, Sasha, Shiry, Tinghui, Weicheng, Yong Jae, Zhe and others, for your guidance, support, discussions, and friendship.

Thank you Angie for shielding me from Berkeley bureaucracy, for organizing many fun events for BAIR and BDD, and for helping make many holidays feel festive with cards, crafting, and conversation.

Thank you Marisa, Emily, Rudy, Steph, Rob, and all of my Cal club water polo teammates. Playing water polo with you has been one of the greatest joys of my life. Go Bears.

Thank you Heather. Your dataset labels were clutch. Thank you for cooking and delivering delicious food to me in lab during times of deadline. Thank you for introducing me to the two best stress-relief devices ever invented: 1) CrossFit, and 2) Donkey Kong Country.

Thank you family and friends for your support and kind encouragement.

Chapter 1

Introduction

People are everywhere! There are roughly 7.7 billion people in the world, and by nature, human beings are social, constantly interacting with each other. Humans are all the time anticipating how others will react and respond to their own actions, and these anticipations help shape daily interactions. Providing machines with the ability to understand and reason about human behavior is arguably one of the most important, intriguing, and potentially useful challenges of modern engineering. Applications are ubiquitous, including automating surveillance, helping care for the elderly or disabled, self-driving cars, and providing sports coaching. As Kenneth Craik famously wrote in 1943, an organism that carries a small-scale model of reality and its own possible actions is better prepared to react to future situations.

To that end, in this thesis, we present methods for learning to predict a variety of human behaviors from video. Broadly, our predictions are at three levels of granularity: events, motion trajectories, and dynamics. In general, structured environments, such as roads, subway stations, airports, restaurants, gyms, and sports matches provide good settings for developing methods to understand and predict human behavior. Their spatio-temporal data has many useful properties that make them favorable settings for studying human behavior. Most importantly, behaviors in these environments are well-defined.

In the first two parts of this thesis, we focus on the sports context. The sports setting is not at all contrived. People all over the world like to watch, play, analyze, and predict sports matches. Of the top ten most popular sports in the world, association football (soccer) is the most popular with an estimated 3.5 billion fans. And, American football is the least popular with an estimated only 400 million fans. Not to mention, over the last year, many states in the United States have legalized online sports betting, so accurate sports predictions could become lucrative.

This vast interest in sports across the world has helped motivate research on sports analytics and various sports-related computer vision problems. Prior to recent advances, and this was only within the last twenty years, players were recruited for major league baseball according to how much they *looked like a baseball player*. Scouts would often go with their *gut feeling* on a recruit, which was primarily based on the players athleticism. The failure of Billy Beane to reach his predicted potential, along with the limited budget of the Oakland As helped spark a revolution in baseball analytics, where the As began recruiting players based on their ability to get on base. Instead of buying players, the As focused on buying runs. Their innovative strategy led to an historic twenty consecutive wins in their 2002 season.

Until recently, most analyses of players and teams were based on easily-measurable statistics, such as field goal percentage or number of assists. These statistics give a limited view of a players game play. For example, in the 1995/96 NBA season, Michael Jordan scored 42.7% of his attempts from three point range. A year later, he only scored 37.4%. Did he become a worse long-range shooter? Or was he just more closely guarded, and hence taking more difficult shots? These are questions that cannot be answered without more information, such as player trajectories.

Eight years ago, the NBA started recording player and ball point trajectories in an overhead, court-space. The availability of player and ball tracking data has led to many new analyses. Including, identifying where on the court individual players are efficient shooters, methods for automatically recognizing ball screens and the defenses response to a screen, automatic play retrieval, and an imitation learning-based approach for developing coordinated movement policies for the players on defense.

From a computer vision perspective, in sports videos, player detection and tracking has long been a favorite and particularly challenging application domain, as players move quickly, undergo frequent occlusions, and wear similar-looking uniforms. One of the classic tracking techniques, mean shift tracking [28], was first demonstrated on American football players. Sports have also been a popular setting for action recognition with datasets such as UCF 101 [115], UCF Sports [109], Stanford Olympics [94]. More recently, attention has shifted to more fine-grained tasks, such as recognizing specific basketball shots and identifying the key players performing those shots [104], and social scene understanding [11].

Video analysis is an active research area. A large body of work has focused on action recognition [17, 141, 73, 115, 19, 113], people and object tracking [121, 135, 142], forecasting pedestrian trajectories [67, 70, 62, 66, 107] and anticipating future human activities [66, 68, 125, 151, 72, 52]. Other works have explored predicting future world states from single images [39, 128, 93], predicting pixel values of future frames [90, 105] and learning dynamical models of objects [39, 40]. These works primarily focus on single-agent events. Whereas, predicting events in sports requires prediction in multi-agent environments that involve adversarial human-human interactions. Some recent work has focused specifically on prediction in the sports context. For instance, [134] use tracks of ball trajectories in tennis games to predict where the ball would be hit. While [143] introduces a collaborative filtering-based method to predict whether a player will hold, pass, or shoot the ball. And others propose the use of hidden conditional random fields for predicting which player will receive the ball next in soccer games, using ground truth trajectories and an encoding of the game state [133].

In many sports, the movement of the ball determines the game outcome. The team that scores the most wins! The ball is the single most import object in play at any moment of a game. And so, in our work, we initially focus directly on the most important question during the game: where will the ball go next? Developing methods to address this question, and the more general question of what event will happen next is the focus of Chapter 2. In Chapter 3, we present a system that is capable of personalized prediction of the future motion of multi-agent, adversarial interactions, again in the sports domain. Finally, in Chapter 4, we cast aside our structured sports environment, and we present a framework for learning a representation of human dynamics in any video that we can: 1) use to estimate the 3d pose and shape of people moving in videos, and 2) use to hallucinate the motion surrounding a single-frame snapshot. There we demonstrate the value of pseudo-ground truth labels derived from Internet videos. We conclude with several promising future directions for learning to predict human behavior from video.

Chapter 2

Discrete Prediction: Predicting Events

The ability of an algorithm to recognize an event does not imply understanding. For example, an algorithm may easily distinguish between a three point shot versus a layup event in basketball, without truly needing to understand anything about the game of basketball itself, beyond of course that there exist various shot types. While there is not one precise, agreed-upon definition for *understanding*, one way of measuring a system's understanding is by evaluating its ability to predict the future. In this chapter, we present various methods for predicting discrete events. We develop these methods in the context of the sports domain, where vast amounts of data are available, and the events are both many and well-defined.

2.1 Introduction

In 2002, Billy Beane defied conventional wisdom by performing meticulous statistical evaluations of undervalued players to assemble the Oakland Athletics baseball team on a negligible budget. His team made history with a record-setting 20-game win streak, and this tremendous feat is documented in the academy award nominated film *Moneyball*. Their success made an unprecedented case for competitive advantages gained by new analyses of individual players' game play. Now imagine if, in addition to knowing the shot success rate of Stephen Curry, the best basketball shooter to date, it is also possible to predict that he is more likely to attempt a shot within zero, one, and two seconds of a pass when his teammates are in a diamond, ring, and triangle formation, respectively. Such predictions are invaluable to the defending team in planning strategy. Billy Beane's analysis revolutionized strategic thinking in baseball, and similarly, we believe statistical methods for predicting player moves have the potential to impact how teams plan their strategies of play.

Predicting player moves in sports videos is an instance of a much grander research agenda to develop algorithms that can predict future events directly from visual inputs. The ability to forecast is a key aspect of human intelligence. Kenneth Craik famously wrote in 1943, "*If the organism*

This chapter is based on joint work with Pulkit Agrawal and Jitendra Malik [35], presented primarily as it appeared in the ICCV 2017 proceedings.

carries a 'small scale model' of external reality and its own possible actions within its head, it is able try out various alternatives, conclude which is the best of them, react to future situations before they arise and in every way react in much fuller, safer and more competent manner to emergencies which face it." While there has been a lot of interest in this problem [66, 105, 125, 56, 90, 126, 3, 68, 40, 95, 149, 128, 129, 5], we lack a good benchmark for comparing different forecasting algorithms.

For multiple reasons, it appears to us that team sports videos are a very good benchmark for evaluating forecasting algorithms. Firstly, many human activities are social and team sports provide an opportunity to study forecasting in an adversarial multi-agent environment. Secondly, team sports are composed of a large and diverse set of discrete events, such as passing, shooting, dribbling, etc. The sequence of events reflects the game play strategies of the two teams, and thus forecasting requires game specific knowledge combined with other visual cues, such as player pose and game state. This implies that for any system to make accurate predictions directly from visual imagery, it must distill game specific knowledge by crunching large amounts of data. Representing such knowledge is a central problem in forecasting, which is put to test in this setup. Expert players and coaches gain such knowledge via experience gathered over long periods of time. An additional benefit of predicting discrete events is crisp and straightforward evaluation of the information of interest that avoids the problems associated with evaluating pixel-level predictions.

In this work, we present a generic framework for predicting future events in team sports directly from visual inputs, and we introduce water polo and basketball datasets for evaluation. These datasets contain game stream accompanied by annotations of player tracks and seventeen different events. The task of interest is: given a history of observations, predict what event will happen immediately, after 1s, or after 2s. The seventeen events are answers to questions that are of great interest in team sports such as - *where will the ball go next? will the player score? will there be a "screen" event? will there be a block? will there be a turnover? will there be a dribble?* among many others.

We construct two set of models - ones that forecast from the raw video stream without any pre-processing and other that transform the raw video stream into an "overhead" representation where the players and balls are represented as dots on the playing field prior to forecasting. Using the water polo dataset as a case study, we present the entire system to convert images captured from a single moving camera into the overhead representation which is then fed into the predictor. We find that the overhead representation leads to more accurate predictions than raw image based representation. The performance of our system is close to humans but worse than water polo experts. We then apply the same set of forecasting techniques on a dataset of basketball games and show that our system outperforms humans on forecasting events in basketball games. While we present results on water polo and basketball, we make no game specific assumptions. The techniques developed in this work apply to a wide number of other team sports such as hockey, rugby, and soccer.

2.2 Related Work

Video analysis is an active research area. A large body of work has focused on action recognition [17, 141, 73, 115, 19, 113], people and object tracking [121, 135, 142]. In contrast to these works we are interested in the problem of forecasting. Predicting pedestrian trajectories [67, 70, 62, 66, 107] and anticipating future human activities [66, 68, 125, 151, 72, 52] has seen considerable interest over the past few years. However, these works mostly consider predicting events related to a single human, while we attempt to forecast events in multi-agent environments involving adversarial human-human interaction. Other works have explored predicting future world states from single images [39, 128, 93, 40], but have been limited to simulation environments or involve a single agent. Predicting pixel values in future frames has also drawn considerable interest [90, 105] but is limited to very short term predictions.

Sport Video Analysis: Traditional work in computer vision analyzing sports videos [13] has focused on either tracking players [14] or balls [87]. Another body of work assumes the annotations of ball or player tracks to analyze game formations or skill level of individual players. For instance, [134] use tracks of ball trajectories in tennis games to predict where the ball would be hit, [15] analyze soccer matches using player tracks. [84] discover team formation and plays using player role representation instead of player identity. More recently techniques such as [104] have looked at the problem of identifying the key players and events in basketball game videos. Closest to our work is the work of [133] that proposes the use of hidden conditional random fields for predicting which player will receive the ball next in soccer games. They assume the knowledge of game state such as attack, defense, counter attack, free kick etc. and assume that identity of players is known. In contrast, we present a forecasting system that works directly from visual inputs. It either uses images directly or converts them it into an overhead view representation using computer vision techniques. We do not require any external annotations of the game state.

2.3 Datasets

We have focused our efforts on the most popular style of sport, *team goal sports*. We select water polo and basketball as two canonical examples because together they capture many diverse aspects of team goal sports: basketball is fast-moving and high-scoring like hurling and handball, while water polo is low-scoring like soccer and has man-up situations like hockey and lacrosse. Despite the different nuances of each team goal sport, they all share many common "events" during game play. For example, players advance the ball toward the goal themselves in a *drive*, and sometimes this results in a *goal* and other times in a *block* or a *missed shot*. Players *pass* the ball to their teammates, and sometimes the defense intercepts the pass.

Water polo

A water polo game naturally partitions into a sequence of alternating team possessions. During a possession, the attacking team's field players primarily spend time in their *front court*, which

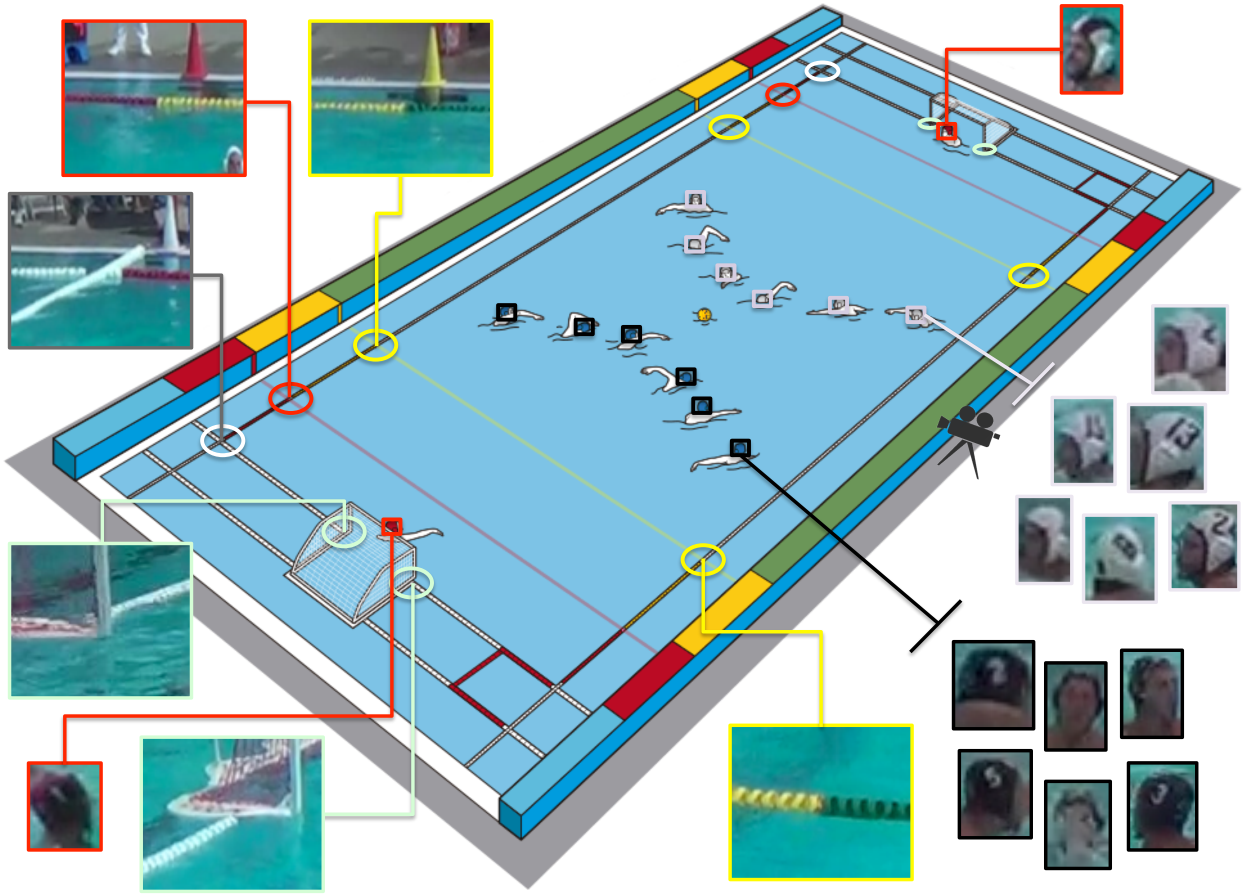


Figure 2.1: Water polo dataset annotations. From single-camera video recordings of 5 water polo games, we collected bounding box track annotations of dark, light, and red-cap player heads. We also collected annotations of pool points marking the field of play: the 2m and 5m lines, the corner of the field, and the points where the cage and lane line meet.

accounts for most of the interesting game play. The attacking team is required to take a shot within 30s, and failure to do so results in a *turnover*. Players of the two teams wear dark colored (typically blue/black) and light colored (typically white) caps. In the remainder of the paper we use dark-cap and light-cap to refer to two teams.

We collected a dataset of front court team possessions from video recordings of 5 water polo games between Division I National Collegiate Athletic Association varsity teams. Similar to the NBA for basketball, this represents the highest level of water polo play in the United States. We chose to focus only on front court possessions, as most interesting events happen during this period. The time intervals of the front court possessions were hand-marked by an expert water polo coach. All the games, four of which are men's games and the other a women's game, are played at the same outdoor pool on different days at times ranging from morning until night; the dataset exhibits a large range of lighting conditions. The games were recorded with a single freely moving camera that pans between each side of the pool with resolution 720p at 25-30fps. Often the camera is adjusted for a new front court possession, resulting in varied camera motions and zooms.

Player and Pool Track Annotations: Bounding box track annotations (Figure 2.1) of dark and light-cap player heads, goalkeepers, and the head of the player in possession of the ball were collected using the VATIC toolbox [124] and Amazon Mechanical Turk. Player possession is defined to begin at the moment a player grasps the ball and ends at the moment that same player passes/shoots the ball or another player steals the ball. Additional annotations of specific points marking the field of play: the 5m line, the 2m line, the pool corner, and the cage posts were obtained. These field markings provide necessary point correspondences between the image view and overhead view of the game, which enable the computation of the player trajectories in the overhead space from the player locations in the image view. For increased data diversity, annotations were collected for 11 quarters of play from 20 quarters available in the 5 games.

Train/Test Splits: The splits were as follows - *train*: 7 quarters, randomly sampled from the first 4 games; *validation*: light-capped team front court possessions in all 4 quarters of the fifth game; and *test*: dark-capped team front court possessions in all 4 quarters of the fifth game. In total, each split has 232, 134, and 171 respective examples of a player passing the ball in a team's front court.

Human Benchmark: For a comparison with human-level accuracy, human annotators were shown every test image taken just before a player loses possession of the ball and were asked to draw a bounding box around the head of the player which they thought would possess the ball next. Nine non-experts and four water polo experts were evaluated. Non-experts had never seen or played a water polo game. In order to account for their inexperience, non-experts were provided all examples used to train computer algorithms along with the ground-truth answer before being asked to make predictions. The experts had all played competitive water polo for at least four years. Expert and non-expert humans accurately predicted the next ball possessor 73% and 55.3% of the time respectively.

Basketball

The dataset is comprised of ground truth (in contrast to water polo, where it is computed) 2D player and ball trajectories, sampled at 25 Hz, in 82 regular-season NBA games obtained using the STATS SportVU system [116], which is a calibrated six-camera setup in every NBA arena. The data includes labels for 16 basketball events, including free throw, field goal, pass, dribble, (player) possession, etc. The data distribution is illustrated in Figure 2.2. As expected, given the nature of the game of basketball, the data is very imbalanced, with 48% of events as dribble, and only 0.24% of events as time out.

Train/Test Splits: A total of roughly $300k$ labeled events were randomly split into $180k$, $30k$, and $90k$ examples for train, validation, and test sets.

Human Benchmark: A set of 18 annotators familiar with basketball were shown a series of fifteen 5-second clips of basketball data, ending with a labeled event. The ball and player trajectories were removed from the final n seconds of the clip, and humans predicted the event at the end of the blanked portion. For each $n \in \{0.4, 1, 2\}$, each human labeled 5 examples randomly sampled

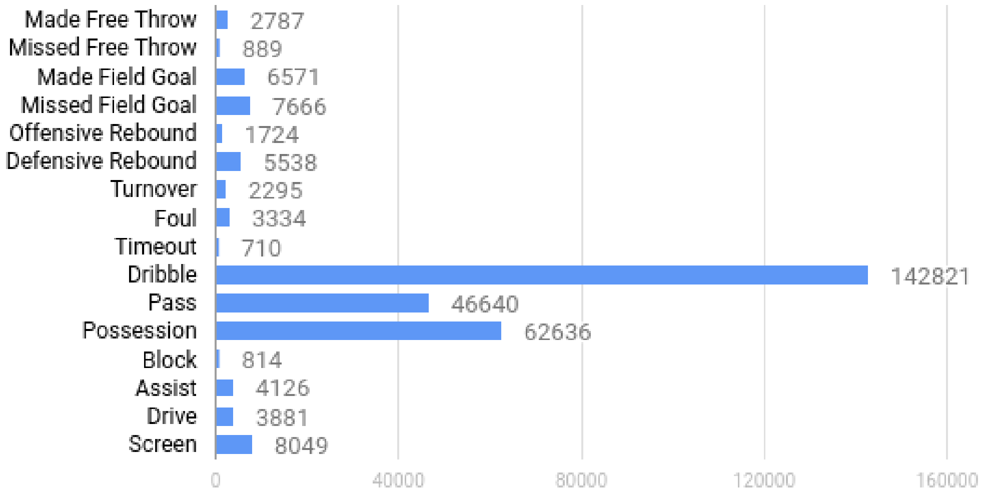


Figure 2.2: Basketball event data distribution, of the 16 labeled events in the basketball dataset, which has over 300k labeled events in total.

from a pool of 80 examples (5 examples of each of the 16 events). Humans were correct 13.5%, 20.6%, and 24.4% for $n = 2$, 1, and 0.4, respectively.

2.4 Methods: From Images to Overhead View

Representing game play with a 2D overhead view of the game, where players are represented as dots at their (x, y) coordinate locations, is often used by coaches. It provides immediate insight into player spacing, and distills player movement into a canonical, simple representation that is easy to compare across many game plays. We construct the overhead representation by first detecting players and the ball. Using knowledge of playing field dimensions and locations of few landmarks, we estimate a homography to transform these detections into a canonical frame. We then link players across frames using tracking. Each step of this process is detailed below.

Player Detection: We finetune a VGG-16 network [112] pre-trained on ImageNet for detecting light and dark cap players using Fast R-CNN [44] and the annotations we collected described in section 2.3. The performance of dark and light color cap person detectors was 73.4% and 60.4%, respectively. We attribute the worse performance of the light-color cap detector to a few confounding factors: 1) many light-color caps were annotated, by one turker, with loose bounding boxes, 2) overhead lights produce specularities, and 3) water splashes can appear visually similar to light-color caps.

Player Tracking: We track by detection. The Hungarian matching algorithm [71] is used to link Fast-RCNN player detections to form player tracks. The pairwise affinity between detections in two sequential frames is a linear combination of Euclidean distance and bounding box overlap.

Overhead Projection: In the case of water polo (Figure 2.3) we used the annotations of 2m and 5m lines, the pool corner, and the cage posts to estimate the homography between the current image captured by the camera and a canonical 2D coordinate system representing the field of play

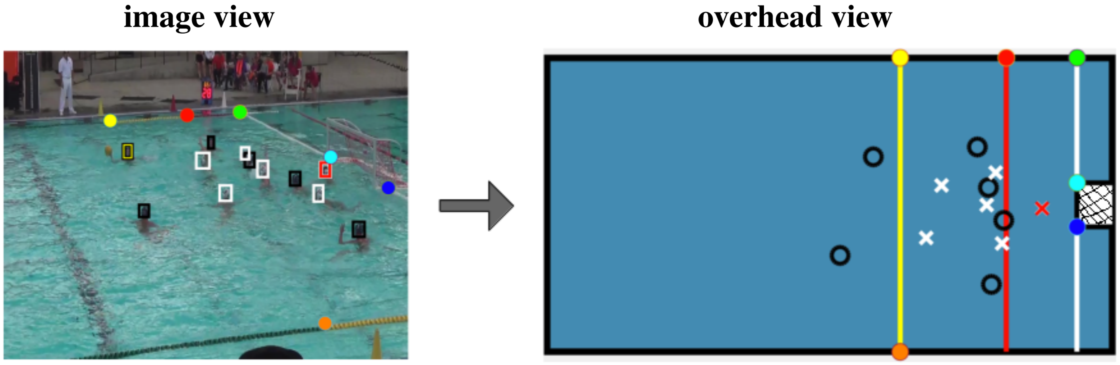


Figure 2.3: The image is converted into the overhead view by first estimating the homography between the image a canonical model of the playing field using field markings such as 2m/5m lines etc. The players are then detected and their locations are transformed using the estimated homography.

using the normalized direct linear transformation (DLT) algorithm [49]. Next, we transform the midpoint of bottom edge of the player bounding box into a (x, y) location in the canonical frame. We use the bottom edge because that is the point of the player that is closest to the field of play, which in turn is mapped to the canonical frame by the homography transformation.

2.5 Experiments

Predicting Future Ball Position

In team goal sports, the movement of the ball determines the game outcome, and hence, it is the single most important object in play at any moment of the game. We, therefore, focus directly on the most important question during the game: where will the ball go next? We study two slightly different variants of this question: In the water polo dataset, we only consider the frame before which the ball possessor is about to lose of the possession of the ball, and we try to predict which player will be in possession of ball next. In the basketball domain, we have access to much more data, and we additionally attempt the more general problem: where will the ball be in one or two seconds in the future?

Water polo: Who will possess the ball next?

In the typical front court water polo scene, there are 6 field players on the attack, defended by 1 goalkeeper and 6 field players on the opposing team. For example, in Figure 2.3, the dark-cap players are on the attack and the light-cap players are on defense. By definition, one of the attacking team players is in possession of the ball. Our system takes as input the frame just before the player loses ball possession by either making a pass to a teammate, shooting the ball, or committing a turnover. The task is to predict which player will possess the ball next.

$F_{[1,2]}$: (x,y) of player with ball

$F_{[3,4]}$: (x,y) of player

$F_{[5,6]}$: (x,y) of nearest defender

F_7: same-team flag

F_8: $||F_{[3,4]} - F_{[1,2]}||_2$

F_9: $||F_{[3,4]} - F_{[5,6]}||_2$

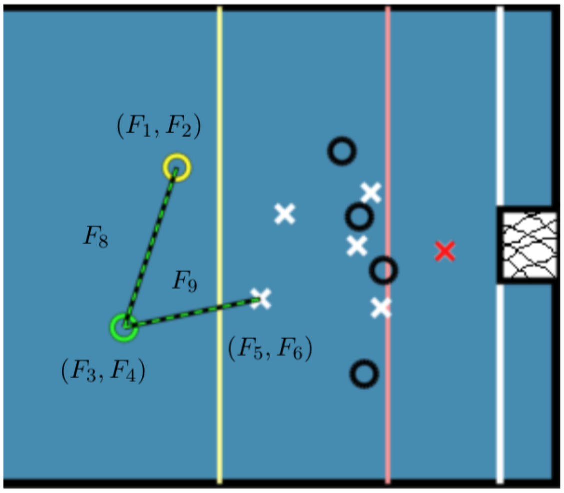


Figure 2.4: The features $F_{[1...9]}$ extracted from the 2D overhead view are used to train a random forest to classify players as either receiving the ball next or not.

A random choice of player from either team would be correct roughly $\frac{1}{12} \approx 8.3\%$ of the time. As a player is more likely to pass the ball to his teammate, a random choice of player from the same team would be correct approximately 20% of the time (empirically validated on the test set). Such random guesses are very naive. Players often tend to pass the ball to nearer teammates, as shorter passes are easier to execute and minimize turnover risk. Predicting the nearest teammate as the next possessor is correct 28.1% of the time. Players also tend to pass the ball to open teammates, those who are not closely guarded by defenders. Predicting a pass to a teammate who is furthest from his nearest defender (i.e. most open) has accuracy of 36.7%. These baselines are considerably worse than an average human with no water polo expertise, who is correct 55.3% of the time.

In the next two sections, we describe how performance can be improved: (1) using additional player features estimated from the overhead representation, and (2) automatically learning feature representations directly from the input image. We operationalize these approaches in the following way: Let there be K players each with feature vector $F^i (i \in \{1, 2.., K\})$, let $b \in \{1, 2.., K\}$ be a discrete random variable that encodes the player in possession of the ball after a pass is made. The goal is to find the player who is most likely to receive the ball, i.e., $\text{argmax}_i \, P(b = i | F^1..F^K)$.

Hand designed features from overhead view

When deciding where to pass the ball, players consider which teammates are in good position to: score, advance the ball, and receive a pass. We formalize these insights and characterize each player using a 9-D feature vector extracted from the overhead representation: the (x, y) player coordinates, the (x, y) coordinates of the nearest player on the opposite team, the (x, y) coordinates of the player in possession of the ball, an indicator flag for whether the player is on the same team as the player in possession of the ball, and the Euclidean distances of the player to the player with the ball and to his nearest defender. This player-centric feature vector is illustrated in Figure 2.4. We assume that features $F^1..F^k$ are mutually independent, and therefore computing $P(b = i | F^1..F^K)$

Method	Ground Truth Heads	Detected Heads
Random, either team	9.5 ± 2.2	9.2 ± 2.2
Random teammate	19.1 ± 3.1	17.0 ± 2.8
Nearest neighbor teammate	28.1 ± 3.4	22.2 ± 3.2
Most open teammate	36.7 ± 3.7	28.7 ± 3.4
$F\,[8 \ldots 9]$	42.5 ± 3.8	35.2 ± 3.6
$F\,[7 \ldots 9]$	45.4 ± 3.4	38.4 ± 4.0
$F\,[3 \ldots 9]$	48.8 ± 4.3	44.1 ± 3.7
$F\,[1 \ldots 9]$	47.1 ± 3.8	45.5 ± 3.5
FCN, teammate	38.1 ± 3.5	35.2 ± 3.6
Human, Non-Expert	55.3 ± 7.9	-
Human, Expert	73.1 ± 2.0	-

Table 2.1: Each row reports accuracy of a different method for predicting which player will possess the ball next. The first four methods are baselines. The intermediate rows provide an ablation study of using various features defined above. The FCN is a deep learning based method and the last two rows report human performance. Performance metrics are reported for two circumstances: using ground truth player locations (column 1) and when detected instead of ground-truth locations (column 2) are used.

reduces to estimating $P(b = i|F^i)$.

We train a system to infer which player will possess the ball next in the following way: we used the pipeline described in section 2.4 to convert the raw image into it's corresponding overhead representation. Next, feature vector of each player was computed from the overhead representation. Finally, a random forest classifier was trained on these features using the training data to estimate $P(b = i|F^i)$. Five-fold cross-validation was performed to chose the optimal depth and number of trees. This system achieved a performance of 45.5% (see Table 2.1) and outperformed the baseline methods on the testing set. Analysis of the results revealed that this method is biased towards predicting the most open perimeter player as the one receiving the ball.

A common failure mode is predicting an open perimeter player, when he is not even facing the player in possession of the ball. These mistakes are not surprising as the overhead view has no access to visual appearance cues. Another possible reason for failures is that the pipeline for converting image data into overhead representation is inaccurate. To tease this apart, we re-ran the analysis using ground truth (instead of estimated) detections. As reported in Table 2.1, the accuracy gap with and without using ground truth detection is within the error bar of the performance on the testing set. This suggests that the pipeline for obtaining overhead representation is accurate and further performance improvements will be gained by building better forecasting models.

Predicting directly from image space

While the overhead view provides a good representation for analyzing game play, it loses subtle visual cues, such as the pose of the player and direction they are facing, that might be very relevant

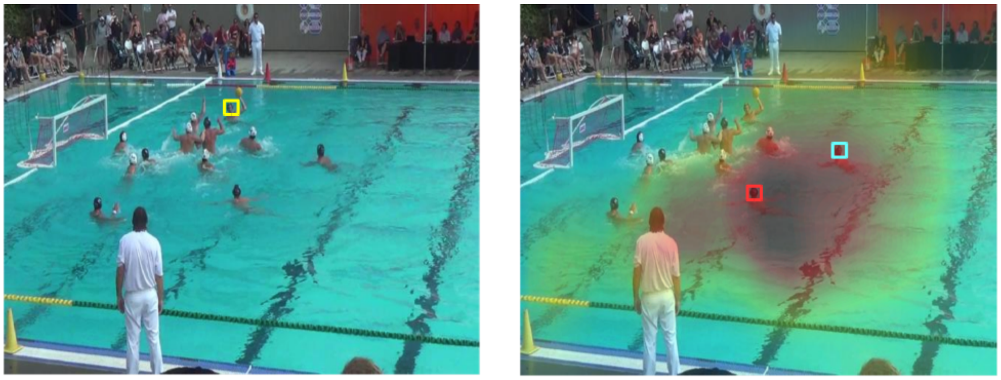


Figure 2.5: The FCN method takes the left image as input and predicts a heatmap (shown overlaid on right) encoding probable ball locations after the ball is passed. The yellow, cyan and red squares indicate the player with the ball, the ground truth player who actually receive the ball next, and the player predicted to receive the ball by the FCN method respectively.

for forecasting. Instead of hand-designing such features, is it possible to automatically discover features that are useful for predicting the next ball possession?

The set of features $F^1..F^k$ is represented by image I_t and we compute $P(b = i|I_t)$ in the following manner: Let l_b, p^k be random variables denoting the future location of the ball and the k^{th} player respectively after the passed ball is received. Since only one player can receive the ball, we assume that if the ball is at location l_b it will be received by the player who has highest probability of presence at l_b (i.e. $\arg\max_k P(p^k = l_b)$). Let l_b^i denote the set of all locations at which $i = \arg\max_k P(p^k = l_b)$. With this,

$$P(b = i|I_t) = \int_{l_b \in l_b^i} P(p^k = l_b, l_b|I_t) \tag{2.1}$$

assuming conditional independence,

$$= \int_{l_b \in l_b^i} P(p^k = l_b|I_t)P(l_b|I_t) \tag{2.2}$$

We model $P(l_b|I_t)$ using a Fully convolutional neural network (FCN; [82]), that takes I_t as input and predicts a confidence mask of the same size as the image encoding $P(l_b|I_t)$. The ground truth value of mask is set to 1 in pixel locations corresponding to bounding box of the player who receives the ball and zero otherwise. The player bounding box is a proxy for future ball location. We finetuned ImageNet pre-trained VGG-16 network for this task.

As we only have 232 training examples, this vanilla system unsurprisingly did not perform well and overfit very easily even with standard data augmentation techniques such as image cropping and dropout regularization. One of our contributions is in showing that the performance can be significantly improved (from 10% to 38.1%) by requiring the FCN system to output the location of players in addition to which player will possess the ball next. Our hypothesis about why this modifications helps is that forcing the CNN to predict player locations results in representations that capture the important feature of player configurations and are thus more likely to generalize

		Non-expert Humans	
		Correct	Incorrect
Random Forest	Correct	32.8	15.8
	Incorrect	22.8	28.6

Table 2.2: Comparing agreement between the predictions of next ball possessor made by humans and our best algorithm on the water polo data. Humans and the algorithm both make correct and incorrect predictions on the same examples more often than not.

than other nuisance factors that the CNN can latch onto given the small size of the training set. This finding is interesting because it suggests that it might be possible to learn even better features by forecasting future player locations for which no additional annotation data is required once the detection and tracking pipeline described in the previous sections is setup.

To estimate $P(p^k = l_b|I_t)$ we first detect all the players in image I_t using the method described in section 2.4. We assume that players will be at the same location after the pass is made. In order to make the ball assignment among players to be mutually exclusive, we use the player locations to perform a Voronoi decomposition of the playing field. Let c^k be the voronoi cell corresponding to the k^{th} player. $P(p^k = l_b)$ is then to set to $\frac{1}{|c^k|}$ if $l_b \in c^k$ and zero otherwise. We then use equation (2) to compute $P(b = i|I_t)$.

This method performs comparably to the baseline that predicts the most open teammate. Visualization in Figure 2.5 shows a dominant pattern with FCN predictions: it consistently places higher likelihood around the perimeter of team in possession of the ball. This is a very sensible strategy to learn because players around the perimeter are often more open and statistical analysis reveals that there are more passes between perimeter players. Given the limited amount of data, the FCN based approach is unable to capture more nuanced aspects of player configurations or more fine grained visual cues such as the player pose.

Comparison to Human Performance

Figure 2.6 compares the predictions of human non-experts against our best performing system. Some common trends are: Non-experts are more likely to incorrectly predict players near the cage. The random forest based method is biased towards predicting the most open perimeter player. A common failure mode is predicting an open perimeter player, when he is not even facing the player in possession of the ball. These mistakes are not surprising as the overhead view has no access to visual appearance cues. We hope that in the future, when more data is available such features can be learnt from the image space representation. Table 2.2 reports agreement statistics between the predictions of our systems and non-expert humans. The fraction of agreement between humans and our system is the highest (32.8% of examples). These numbers suggest that humans and our system have similar biases and are accurate/prone to errors on similar examples.

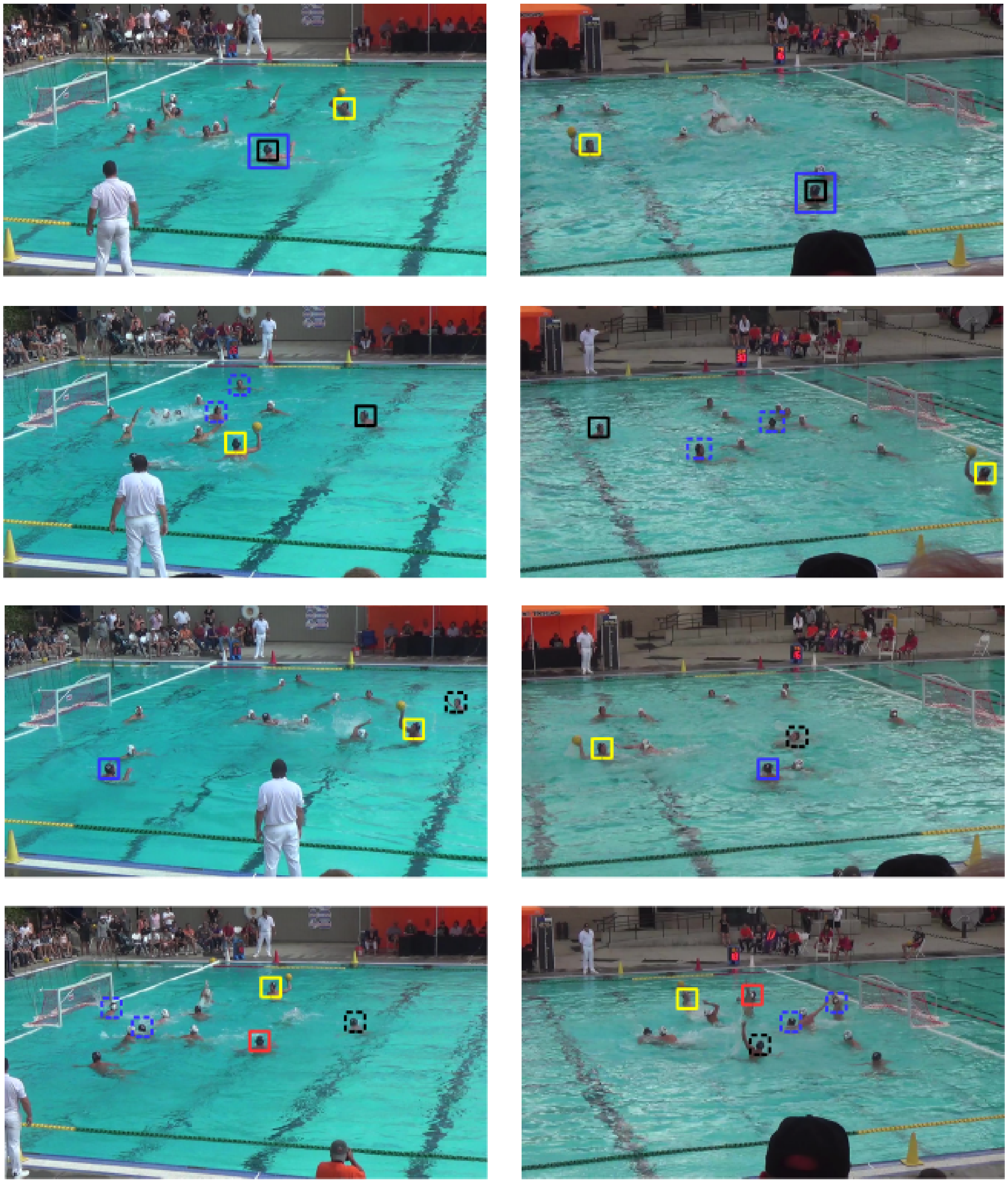


Figure 2.6: Sample predictions of our algorithm (black) and of water polo laymen (blue). The player in possession of the ball is shown in yellow. A solid line indicates a correct prediction. Row 1 shows examples where both were correct. Row 2 shows examples where the algorithm is correct, but humans incorrect. Row 3 shows examples where humans are correct, but our algorithm is incorrect. Finally, row 4 shows examples where both were incorrect, and red indicates the ground truth.

Method	Error (1s in Future)				Error (2s in Future)			
	Distance (%)		Angle (^{o})		Distance (%)		Angle (^{o})	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
Last Position	11.7	10.4	-	-	20.0	18.3	-	-
Ball Velocity	100	100	89.3	88.7	100	100	88.8	85.9
CNN + LSTM	11.4	8.6	61.8	46.6	17.1	14.1	**53.1**	**38.1**
CNN (Early Fusion)	**10.8**	**8.3**	**60.2**	**44.1**	**16.8**	**13.8**	54.3	38.3

Table 2.3: The early fusion CNN outperforms Last Position and Ball Velocity baseline methods and a late fusion CNN based approach in predicting (basket)ball position 1s and 2s in the future. We report mean and median errors in the distance and angle of predicted ball positions.

Basketball: Where will the ball go?

As more data was available for basketball, we attacked the more general problem of predicting where the ball will go next after one and two second respectively. We represented the overhead view as 64x64x3 images where the three channels corresponded to location of players of team 1, players of team 2 and the ball respectively. For capturing temporal context, we included 5 images from the past taken at times $\{t, t-1, ...t-4\}s$ respectively. The task was to predict the ball location at times $\{t+1, t+2\}s$ respectively. To account for multimodality in the output, we formulate this as a classification problem with the xy plane discretized into 256 bins.

We experiment with two different CNN training strategies: (a) early fusion of the temporal information by concatenating 5 images into a 15 channel image that was fed into a CNN or, (b) late fusion by using a LSTM on the output of CNN feature representation of the 5 images. The CNN architecture comprised of 4 convolutional layers containing 32 filters each of size 3x3, stride 2 and ReLU non-linearity. In case of early fusion, the output of the last convolutional layer was fed into a 512-D fully connected layer which in turn fed into the prediction layer. In case of late fusion, the output of the last convolutional layer was fed into a 512-D LSTM layer which in turn fed into a prediction layer. The performance of these networks and some baseline methods is reported in Table 2.3.

We consider two baselines - one which predicts that the ball at time $t+1, t+2$ will remain at the same location as at time t (i.e. Last position). This is a reasonable baseline because in many frames the player is in possession of the ball and he does not move. The second baseline estimates the ball velocity at time t and uses it to forecast the future location. We report mean and median errors in the distance and the angle of prediction. The distance between the ground truth and predicted location is reported as the percentage of the length of the basketball court. The angular error is the angle between the vector 1 pointing from current position to ground truth position in the future and vector 2 pointing from current to predicted position. We find that the proposed methods outperform the baseline and the early fusion method performs slightly better than the late fusion method. As expected, the prediction errors in distance are larger when predicting for 2s as compared to 1s. However, the errors in angle follow the reverse trend. One explanation is that in a shorter period,

Method	Dataset	Δ T	FT made	FT miss	FG made	FG miss	Off. reb.	Def. reb.	Turnover	Foul	Time Out	Dribble	Pass	Poss.	Block	Assist	Drive	Screen	mAP
Avg. Human	H	1s	100.0	0	60.0	12.5	11.1	16.7	0	0	33.3	66.7	0	0	0	0	0	28.6	20.6
Random Forest	H	1s	100.0	0	20.0	0	20.0	40.0	0	0	0	100.0	20.0	40.0	0	0	0	0	21.3
Image CNN	A	1s	46.0	20.3	3.9	4.8	2.0	5.5	0.9	1.6	0	61.7	16.5	22.9	0.9	1.8	1.6	3.7	11.9
Overhead CNN	A	1s	62.2	22.7	38.6	16.4	9.4	43.9	1.8	5.2	3.5	76.1	25.5	37.6	0.8	3.4	1.6	13.1	22.6
Random Forest	A	1s	75.5	41.4	41.3	15.7	11.8	61.2	2.3	5.6	4.5	80.5	26.7	40.9	1.0	3.5	1.2	8.5	26.4
Avg. Human	H	2s	33.3	20.0	14.3	0	0	0	0	0	37.5	75.0	0	0	0	0	16.7	20.0	13.5
Random Forest	H	2s	100.0	0	0	0	20.0	40.0	0	0	0	100.0	0	20.0	0	0	0	0	17.5
Image CNN	A	2s	32.5	7.8	1.9	2.5	0.9	2.7	0.5	0.8	0.2	53.8	14.7	19.9	0	0.6	0.6	2.9	8.8
Overhead CNN	A	2s	39.8	19.0	7.3	6.9	3.8	12.9	1.5	2.2	1.6	71.0	18.3	25.3	0.4	2.7	1.1	5.8	13.7
Random Forest	A	2s	66.9	29.7	11.8	7.3	5.0	35.4	1.5	2.6	2.7	76.4	21.4	30.2	0.3	2.5	0.9	5.0	18.7
Avg. Human	H	40ms	28.6	28.6	83.3	0	50.0	0	0	0	0	25.0	57.1	14.3	0	0	20.0	83.3	24.4
Random Forest	H	40ms	100.0	0	40.0	80.0	40.0	100.0	0	20.0	0	100.0	60.0	100.0	0	0	0	80.0	45.0
Random Forest	A	40ms	68.8	24.5	69.5	54.7	62.7	85.2	6.1	31.8	16.7	93.2	76.2	92.6	3.3	8.1	5.0	57.7	47.3

Table 2.4: Prediction accuracy ΔT seconds in the future of 16 basketball events: free throw (FT) made and missed, field goal (FG) made and missed, offensive (off) and defensive (def) rebound, etc. Methods were evaluated on the full (A) test split of $90k$ events, as well as a smaller, 80-example subset (H) for human performance evaluation and comparison.

the ball moves by small distances and therefore angle measures are not robust.

Transferring from Basketball to Water polo

Basketball and water polo are both team sports that require scoring baskets/goals. This suggests that there maybe general game play strategies, e.g., pass to the most open player, that are shared between these two games. If this is indeed the case then a model trained on one of these sports should perform reasonably well on forecasting events in the other sport. In order to test this hypothesis we trained a random forest model on the basketball data (the larger dataset) for predicting which player will get the ball next using the same features as described in 2.5 and then tested it on the water polo testing set.

The accuracy of this model on basketball itself was 69.9% and 36.8% on water polo. The performance on water polo is worse than a model trained directly on water polo (which achieves 45.5%) but same as the most open teammate baseline with 36.7% accuracy (Table 2.1). One explanation of these results is that differences in game strategies arise from the differences in game rules, number of players, and field size. Therefore the basketball model is outperformed by a model trained on water polo itself. However, the transfer performance is significantly better than chance performance and nearest teammate baseline, suggesting that our method is capable of learning game-independent ball passing strategies. A more detailed analysis of the error modes is provided in the supplementary materials.

Predicting Events in Basketball

Predicting the ball location is just one out of many events of interest. For example, whether a teammate would screen or whether dribble or a break would take place are of great interest in basketball. In a manner similar to predicting where the ball will be at times $\{t+1, t+2\}s$, we predict which out 16 events of interest will happen in the future.

We evaluate random forest and neural network based approaches for this task. The input to the random forest are the following hand designed features, extracted from the last visible frame: player and ball coordinates and velocities, distances between each player and the ball, angles between each player and the ball, the time remaining on the shot clock, the remaining game time in the period, and the time since the most recent event for each event occurring in the visible history. In total, we used 92 features. We tested two different neural networks - (a) Overhead CNN that took as inputs the image representation of the overhead view (see Section 2.5) along with the hand designed features described above and (b) Image CNN that took as input raw RGB images.

Network architecture: The input to the neural network was a tuple of the image representation of the overhead view and the hand designed features. The image representation was input to a CNN with stacked convolutional layers fed into a 256-D fully connected layer, and the hand designed features were input directly to two stacked fully connected layers of dimension 512 and 256, respectively. The 256-D fully connected layer from the CNN and the 256-D fully connected layer from the hand designed features were concatenated and input to the 16-D prediction layer. The CNN architecture we used comprised of 4 convolutional layers containing 32, 64, 64, 32 filters, respectively. The spatial extent of the filters was 3x3. A 3x3 max pooling layer was after the first convolutional layer, and a 2x2 max pooling layer was after the second and fourth convolutional layers. Rectified Linear Units (ReLU) non-linearity was used after each convolutional layer. The final convolutional layer was fed into a 256-D fully connected layer.

Training details: As shown in Figure 2.2, the basketball data is severely imbalanced. We addressed this imbalance by weighting the cross-entropy loss of the network with the inverse class-proportion. We used a batch size of 128, and divided the class weights with a constant (16) that gave the average class weight of 1 in a batch.

Table 2.4 reports the performance of humans and various methods described above at predicting player moves 1s, 2s and 40ms in advance. The two test splits, "*H*" and "*A*" correspond to 80 examples on which humans were evaluated and a set 90K examples on which the algorithm was evaluated. The purpose of reporting the accuracy when predicting 40ms in advance is to obtain an upper bound on performance. The results reveal that random forest outperforms CNN based approaches and both these approaches perform better than an average human. The Overhead CNN outperforms the Image CNN suggesting that extracting features relevant for forecasting from raw visuals is a hard problem. It is also noteworthy that humans are significantly better at identifying Field Goals (i.e., FG made), but worse at identifying other events.

2.6 Discussion

In this work we present predicting next players' moves in basketball and water polo as benchmark tasks for measuring performance of forecasting algorithms. Instead of forecasting activities of a single human, sports require forecasting in adversarial multi-agent environments that are a better reflection of the real world. As the events we predict are discrete, our benchmark allows for a crisp and meaningful evaluation metric that is critical for measuring progress. We compare the performance of two general systems for forecasting player moves: 1) a hand-engineered system that takes raw visuals as inputs, then transforms them into an overhead view for feature extraction, and 2) an end-to-end neural network system. We find the hand-engineered system is close to (non-expert) human performance in water polo and outperforms humans in basketball. In both cases it outperforms the neural network system, which raises a very interesting question - what auxiliary tasks/unsupervised feature learning mechanisms can be used to improve prediction performance. We find that a system trained on basketball data generalizes to water polo data, showing that our techniques are capable of extracting generic game strategies.

Chapter 3

Continuous Prediction: Predicting 2D Motion

In the previous chapter, we focused on methods for predicting discrete events in the context of the sports domain. However, sports, and really life, is about more than just sequences of events. In this chapter, we present methods for predicting human motion, again in the sports context, where data is abundant. Learning to predict sports player motion is also very interesting behavior-wise because by default many sports present a multi-agent, adversarial environment. Furthermore, with different player positions, it also affords the ability to personalize prediction in a meaningful, and interpretable way.

3.1 Introduction

Humans continuously anticipate the future states of their surroundings. Someone extending a hand to another is likely initiating a handshake. A couple entering a restaurant is likely looking for a table for two. A basketball player on defense is likely trying to stay between their opponent and the basket. These predictions are critical for shaping our daily interactions, as they enable humans to navigate crowds, score in sports matches, and generally follow social mores. As such, computer vision systems that are successfully deployed to interact with humans must be capable of forecasting human behavior.

In practice, deploying a computer vision system to make a fine-grain prediction is difficult. Intuitively, people rely on context to make more accurate predictions. For example, a basketball player may be known to stay back in the lane to help protect the rim. The ability to leverage specific information, or *personalize*, should improve the prediction of fine-grained human behavior.

The primary challenge of personalizing prediction of multi-agent motion is to develop a representation that is simultaneously robust to the number of possible permutations arising in a situation and sufficiently fine-grained, so the output prediction is at the desired level of granularity. One typ-

This chapter is based on joint work with Patrick Lucey and Sujoy Ganguly [36], presented primarily as it appeared in the ECCV 2018 proceedings.

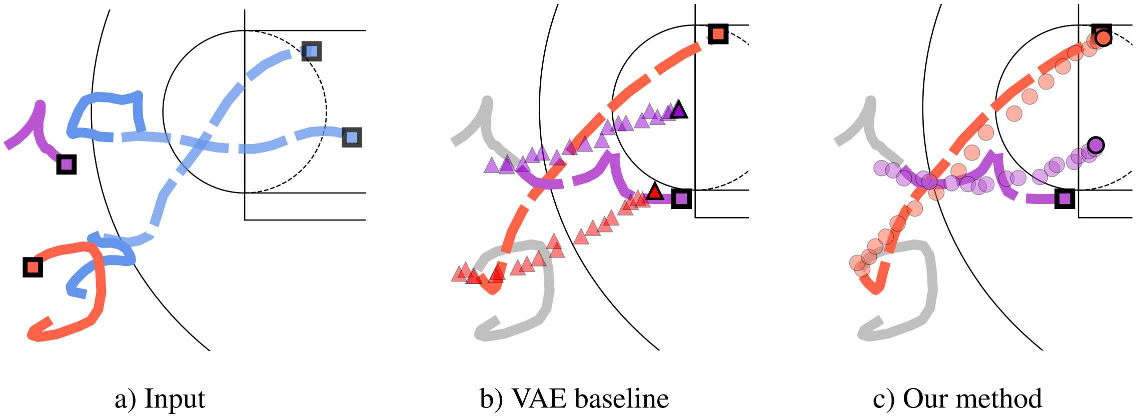

a) Input b) VAE baseline c) Our method

Figure 3.1: a) Given a 2D trajectory history of moving agents (solid lines), and the future motion of a subset of the agents (blue dashed lines); our **prediction task** b) is to generate the most likely motion of the other agents (orange, purple dashed lines). Standard approaches are unable to capture the influence of the group motion (triangles). c) Our method improves performance by incorporating context-specific information (circles).

ically employees one of two approaches: i) *bottom-up* – where each trajectory has the same model applied to it individually, or ii) *top-down* – where a group representation of all trajectories has one model applied to it all at once. The data and target application mainly drive the choice of approach. Typically, in settings with a variable number of agents, *e.g.*, autonomous vehicles or surveillance, one uses a bottom-up approach [76, 6, 58]. When the number of agents is fixed, *e.g.*, sports, faces, and body pose one prefers a top-down approach [4, 85, 54, 75].

While efficient for heavily structured problems, current top-down methods cannot incorporate the necessary context to enable *personalized* prediction, and often require pre-computing some heuristic group representation. Whereas, bottom-up approaches can personalize via a large refinement module [76]. In this paper, we show that by using a conditional variational autoencoder (CVAE), we can create a generative model that simultaneously learns the latent representation of multi-agent trajectories and can predict the agents' context-specific motion.

Due to the vast amount of data available and its adversarial, multi-agent nature, we focus on predicting the motion paths of basketball players. Specifically, we address the problem of forecasting the motion paths of players during a game (Fig. 3.1a). We demonstrate the effectiveness of our approach on a new basketball dataset consisting of sequences of play from over 1200 games, which contains position data of players and the ball.

To understand the function of initial data representation, context, personalization of agent trajectory prediction and generative modeling, we divide our problem into three parts. First, to understand the role of data representation on prediction, we predict the offense given the motion history of all players (Fig. 3.1b). By applying *alignment* to the multi-agent trajectories we minimize the problem of permutation allowing our group representation of player motion to outperform the current state of the art methods. Next, to understand the role of context, we compare the prediction of offensive agents given the motion of the defense, player and team identities. We use separate encoders for context and player/team identity which we connect to the variational layer, as opposed to being used in a ranking and refinement layer, and thus act directly as conditionals. By conditioning on context with alignment and identity, we can generate a very accurate, fine-grained, prediction of any group of agents without the need for an additional refinement module (Fig 3.1c). Finally, we tackle the challenge of forecasting the motion of subsets of players (a mixture of offense and defense), given the motion of the other remaining players. Again we find that our CVAE far outperforms the previous state of the art methods by a factor of two and that it can make reasonable predictions given only the motion history and the player and team identities when predicting the future motion of all ten players.

3.2 Related Work

Forecasting Multi-Agent Motion Lee et al. [76] provide an excellent review of recent path prediction methods, in which they chronicle previous works that utilize classical methods, inverse reinforcement learning, interactions, sequential prediction and deep generative models. For predicting multi-agent motion paths, there are two primary bodies of work: *bottom-up* and *top-down* approaches.

Regarding bottom-up approaches, where the number of agents varies, Lee et al. [76] recently proposed their DESIRE framework, which consisted of two main modules. First, they utilized a CVAE-based RNN encoder-decoder which generated multiple plausible predictions. These predictions, along with context, were fed to a ranking and refinement module that assigns a reward function. The predictions are then iteratively refined to obtain a maximum accumulated future reward. They demonstrated the approach on data from autonomous vehicles and aerial drones and outperformed other RNN-based methods [58]; however, in the absence of the refinement module, the predictions were poor.

For predicting variable numbers of humans moving in crowded spaces, Alahi et al. [6] introduced the idea of “Social LSTMs” which connected neighboring LSTMs in a social pooling layer. The intuition behind this approach is that instead of utilizing all possible information in the scene, the model only focuses on people who are near each other. The model will then learn that behavior from data, which was shown to improve over traditional approaches which use hand-crafted functions such as social forces [139]. Many authors have applied similar methods for multi-agent tracking using trajectories [21, 132, 86].

Nearly all work that considers multiple agents via a top-down approach is concerned with modeling behaviors in sports. Kim et al. [63] used the global motion of all players to predict the future location of the ball in soccer. Chen et al. [27] used an occupancy map of noisy player detections to predict the camera-motion of a basketball broadcast. Zheng et al. [147] used an image-based representation of player positions over time to simulate the future location of a basketball. Lucey et al. [85] learned role representations from raw positional data, while Le et al., [75] utilized a similar representation with a deep neural network to imitate the motion paths of an entire soccer team. Felsen et al. [35] used hand-crafted features to predict future events in water polo and basketball.

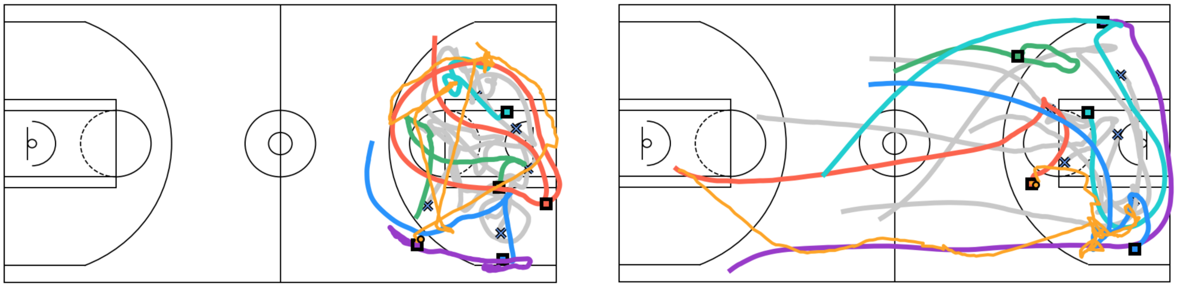


Figure 3.2: **Dataset** Example plays from our basketball dataset, which contains 95,002 12-second sequences of offense (color), defense (gray), and ball (orange) 2D overhead-view trajectories. The identity, team, and canonical position of each player are known.

Lastly Su et al. [117] used ego-centric appearance and joint attention to model social dynamics and predict the motion of basketball players. In this paper, we utilize the representation which most closely resembles Le et al. [75], the CVAE approach utilized by [76], and a prediction task similar to [117].

Personalization to Tracking Data Recommendation systems, which provide *personalized* predictions for various tasks often use matrix factorization techniques [69]. However, such techniques operate under the assumption that one can decompose the data linearly, using hand-crafted features to capture the non-linearities. However, in conjunction with deep models and the vast amount of vision data, recommendation engines based on vision data are starting to emerge. Recently, Deng et al. [32] used a factorized variational autoencoder to model audience reaction to full-feature length movies. Charles et al. [26] proposed using a CNN to personalize pose estimation to a person's appearance over time. Insafutdinov et al., [54] used a graph partitioning to group similar body-parts to enable effective body-pose tracking. In all of these works, they use their deep networks to find the low-dimensional embedding at the encoder state which they use to personalize their predictions. In this work, we followed a similar strategy but included the embedding in a variational module.

Conditional Variational Autoencoders Variational Autoencoders [64] are similar to traditional autoencoders, but have an added regularization of the latent space, which allows for the generation of new examples in a variety of contexts [46, 18]. Since the task of fine-grained prediction is naturally one in which history and context determine the future motions, we utilize a conditional variational autoencoder (CVAE) [65, 114]. In computer vision, CVAEs have recently been used for inpainting [97, 98], and for predicting the future motion of agents in complex scenes [76, 129]. In this paper, we apply the idea of conditioning on the history and the surrounding context to predict the personalized adversarial motion of multiple agents without ranking or refinement.

3.3 Basketball Tracking Dataset

Team sports provide an ideal setting for evaluating personalized behavior models. Firstly, there is a vast amount of labeled data in sports, including potentially thousands of data points for each player. Furthermore, the behaviors in team sports are well-defined and complex, with multiple agents simultaneously interacting collaboratively and adversarially. Therefore, sports tracking data is a good compromise between completely unstructured tracking data (*e.g.*, pedestrian motion where the number of agents is unconstrained) and highly structured data (*e.g.*, body pose or facial tracking where the number of agents is both fixed and physically connected). To that end, we present basketball as a canonical example of a team goal sport, and we introduce a new basketball dataset.

Our proposed dataset is composed of 95,002 12-second sequences of the 2D basketball player and ball overhead-view point trajectories from 1247 games in the 2015/16 NBA season. The trajectories are obtained from the STATS in-venue system of six stationary, calibrated cameras, which projects the 3D locations of players and the ball onto a 2D overhead view of the court. Fig. 3.2 visualizes two example sequences. Each sequence, sampled at 25 Hz, has the same team on offense for the full duration, ends in either a shot, turnover or foul. By eliminating transition plays where teams switch from defense to offense mid-sequence, we constrain the sequences to contain persistent offense and defense. Each sequence is zero-centered to the court center and aligned, so the offense always shoots toward the court's right-side basket. In our experiments, we subsample the trajectory data at 5 Hz, thereby reducing the data dimensionality without compromising information about quick changes of direction.

Personalization We label each sequence with its player identity, team, canonical position (*i.e.*, point/shooting guard, small/power forward, center), and aligned position (Section 3.4). Only the 210 players with the most playing time across all sequences are assigned unique identities. The remaining players are labeled by their canonical position, thus limiting the set of player identities.

Data splits The data is randomly split into train, validation, and test sets with 60708, 15244, and 19050 sequences in each respective split.

3.4 Methods

We frame the multi-agent trajectory prediction problem as follows: In a 2D environment, a set A of interacting agents are observed over the time history $[t_0, t_q]$ to have trajectories $X_A^{[t_0,t_q]} = \{X_i^{[t_0,t_q]}\}|_{\forall i \in A}$. The trajectory history of the i^{th} agent is defined as $X_i^{[t_0,t_q]} = \{x_i^{t_0}, x_i^{t_0+1}, \cdots, x_i^{t_q}\}$, where x_i^t represents the 2D coordinates of the trajectory at time t. We wish to predict the subsequent future motion, to time t_f, of a subset of agents $P \subseteq A$. In other words, our objective is to learn the posterior distribution $P(Y_P^{(t_q,t_f]} | X_A^{[t_0,t_q]}, O)$ of the future trajectory motion of the agents in subset P, specifically $Y_P^{(t_q,t_f]} = \{Y_j^{(t_q,t_f]}\}|_{\forall j \in P}$.

In addition to the observed trajectory history, we also condition our learned future trajectory distribution on other available observations O. In particular, O may consist of: 1) the identities ϱ

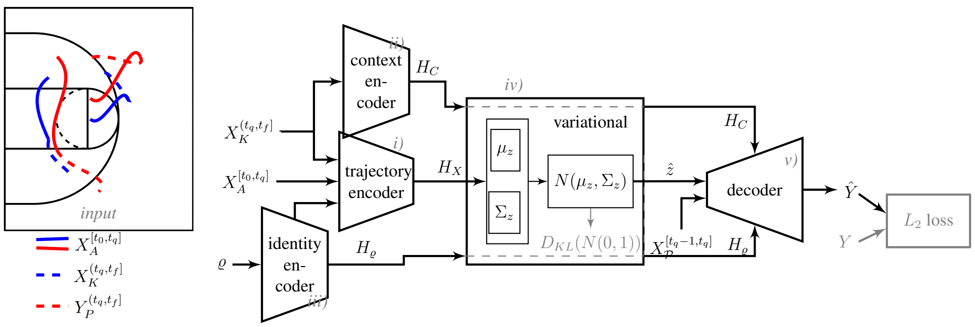


Figure 3.3: **Model architecture.** The inputs to the *i)* trajectory encoder are the tracking history of all players $X_A^{[t_0,t_q]}$, the identity ϱ, and the context $X_K^{(t_q,t_f]}$. The trajectory context $X_K^{(t_q,t_f]}$ is *ii)* encoded as H_C. The one-hot-encoded player or team identity ϱ is *iii)* encoded as H_ρ. The *iv)* variational module predicts the mean μ_z and standard deviation Σ_z of the latent variable distribution $\mathcal{N}(\mu_z, \Sigma_z)$. A random sample $\hat{z}$ from $\mathcal{N}(\mu_z, \Sigma_z)$ is input to the decoder, along with the conditionals H_C, H_ρ, and the last one second of of player motions $X_{\mathcal{A}}^{[t_q-fps,t_q]}$. The *v)* decoder then predicts the future paths $\hat{Y}$. At train time the KL divergence and L_2 loss are minimized.

of the agents in P, and 2) the future context C, represented by the future trajectories $X_K^{(t_q,t_f]} = \{X_\ell^{(t_q,t_f]}\}|_{\forall \ell \in K}$ of agents in the set $K \subset A$ s.t. $K \cup P = A$, $K \cap P = \{\}$. One of the main contributions of this work is how to include various types of information into O, and the influence of each information type on the prediction accuracy of $Y_P^{(t_q,t_f]}$ (Section 3.5).

The conditionals and inputs to our model are each encoded in their encoders. To learn the posterior, we use a CVAE, which allows for the conditional generation of trajectories while modeling the uncertainty of future prediction. In our case, the CVAE learns to approximate the distribution $P(Y_P^{(t_q,t_f]} \mid X_A^{[t_0,t_q]}, O)$ by introducing a random D_z-dimensional latent variable z. The CVAE enables solving one-to-many problems, such as prediction, by learning a distribution $Q(z = \hat{z} \mid X_A^{[t_0,t_q]}, O)$ that best reconstructs $Y_P^{(t_q,t_f]}$.

Fig. 3.3 shows our overall model architecture, which is divided into the five modules: i) the trajectory encoder with $X_A^{[t_0,t_q]}$ and O as input, ii) the context encoder with $X_K^{(t_q,t_f]}$ as input, iii) the identity encoder with ϱ as input, iv) a variational module, and v) the trajectory decoder with sampled latent variable $\hat{z}$ and encoded conditionals as input. The input to the variational module is the joint encoding of the trajectory history $X_A^{[t_0,t_q]}$ with the context and identity. The trajectory history, context, and identity serve as our conditionals in the CVAE, where the context and identity are each separately encoded before being concatenated with $\hat{z}$ as input to the decoder. The trajectory history *conditional* $X_{\mathcal{P}}^{[t_q-1,t_q]}$ for $\hat{z}$ is the last one second of observed trajectory history of the agents in $\mathcal{P}$. This encourages the model predictions to be consistent with the observed history, as our decoder outputs $X_{\mathcal{P}}^{[t_q-1,t_q]}$ concatenated with $Y_P^{(t_q,t_f]}$.

Training phase

We have modeled the latent variable distribution as a normal distribution

$$\begin{aligned} Q\left(z = \hat{z} \mid X_A^{[t_0,t_q]}, X_K^{(t_q,t_f]}, \varrho\right) &= Q\left(z = \hat{z} \mid H_x, H_C, H_\varrho\right) \\ &\sim N\left(\mu_z, \Sigma_z\right). \end{aligned} \tag{3.1}$$

Therefore, at train time the variational module minimizes the Kullback-Leibler (KL) divergence (D_{KL}) and the trajectory decoder minimizes Euclidean distance $\left\|Y - \hat{Y}\right\|_2^2$. For simplicity, let $Y = (X_{\mathcal{P}}^{[t_q-1,t_q]}, Y_P^{(t_q,t_f]})$. The total loss is

$$L = \left\|Y - \hat{Y}\right\|_2^2 + \beta D_{KL}(P||Q), \tag{3.2}$$

where $P\left(z \mid X_A^{[t_0,t_q]}, X_K^{(t_q,t_f]}, \varrho\right) = N(0,1)$ is a prior distribution and β is a weighting factor to control the relative scale of the loss terms. We found that for $\beta = 1$, our model without the conditionals (VAE) would roughly predict the mean trajectory, whereas when $\beta \ll 1$ we were able to predict input-dependent motion. In our proposed model, we observed that $\beta = 1$ performed as well as $\beta \ll 1$, so in all our experiments except for the vanilla VAE, we use $\beta = 1$.

Testing phase

At test time, the input into the trajectory encoder is the trajectory history of all agents $X_A^{[t_0,t_q]}$, the future trajectories of the agents not predicted $X_K^{(t_q,t_f]}$, and the encoded agent identities ϱ. The variational module takes the encoded trajectory H_X, which is also conditioned on the context $X_K^{(t_q,t_f]}$ and the player identities ϱ, and returns a sample of the random latent variable $\hat{z}$. The trajectory decoder then infers the tracks of the agents to be predicted $Y_P^{(t_q,t_f]}$ given a sampled $\hat{z}$, the encoded context H_C, the encoded identities H_ϱ, and the final one second of trajectory history for the agents to be predicted, $X_{\mathcal{P}}^{[t_q-1,t_q]}$.

Trajectory alignment

The network inputs are a concatenation of each 2D agent trajectories. For example, the input $X_A^{[t_0,t_q]}$ forms an $|\mathcal{A}| \times (t_q \cdot 5) \times 2$ array, where $|\mathcal{A}|$ is the number of agents, $t_q \cdot 5$ is the total number of temporal samples over t_q seconds sampled at 5 Hz. One of the significant challenges in encoding multi-agent trajectories is the presence of permutation disorder. In particular, when we concatenate the trajectories of all agents in $\mathcal{A}$ to form $X_A^{[t_0,t_q]}$, we need to select a natural and consistent ordering of the agents. If we concatenate them in a random order, then two similar plays with similar trajectories will have considerably different representations. To minimize the permutation disorder, we need an agent ordering that is consistent from one play to another.

If we have a variable number of agents, it is natural to use an image-based representation of the agent tracks. In our case, where we have a fixed number of agents, we instead align tracks using a tree-based role alignment [110]. This alignment has recently been shown to minimize reconstruction error; therefore it provides an optimal representation of the multi-agent trajectories.

In brief, the tree-based role alignment uses two alternating steps, *i*) an Expectation-Maximization (EM) based alignment of agent positions to a template and *ii*) K-means clustering of the aligned agent positions, where cluster centers form the templates for the next EM step. Alternating between EM and clustering leads to a splitting of leaf nodes in a tree until either there are fewer than M frames in a cluster or the depth of the tree exceeds D. For our experiments we used $D = 6$ and trained separate trees for offense $(M = 400)$ and defense $(M = 4000)$. To learn a per-frame alignment tree, we used 120K randomly sampled frames from 10 NBA games from the 2014/15 season.

Implementation details

Architecture All encoders consist of N fully connected layers, where each layer has roughly half the number of units as its input layer. We experimented with different input histories, prediction horizons, and player representations, so we dynamically set the layer structure for each experiment, while maintaining 64 and 16 units in the final layer of the trajectory and context encoders, respectively. For the identity encoder, the final output size depended on the identity representation ϱ, which was either: 1) a (concatenated) one-hot encoding of the team(s) of the players in P (output dimension 5 for single team and 16 for mixed), and 2) a (concatenated) one-hot encoding of

History (s)	Horizon (s)	Trajectory encoder	Context encoder	Latent dimension	Decoder
1	4	200-100-64	200-100-64-32-16	32	200-300
2	4	200-100-64	200-100-64-32-16	32	200-300
4	4	400-200-100-64	400-200-100-64-32-16	32	200-300
6	4	600-400-200-100-64	600-300-150-64-32-16	32	200-300
8	4	600-400-200-100-64	600-300-150-64-32-16	32	200-300
4	1	400-200-100-64-32	400-200-100-64-32-16	16	115-120
4	2	400-200-100-64	400-200-100-64-32-16	32	140-180
4	4	400-200-100-64	400-200-100-64-32-16	32	200-300
4	6	400-200-100-64	400-200-100-64-32-16	32	150-280-420
4	8	400-200-100-64	400-200-100-64-32-16	32	150-300-540

Table 3.1: Fully connected layer dimensions for network architectures designed for offense prediction as a function of history (Table 2 of the main paper) and horizon (Figure 7 of the main paper). For these architectures, the team identity encoder has layers with dimension 20-10-5. Each layer has elu activation, except the final layer in the decoder, which has sigmoid activation.

each player identity in P.

Network architectures Table 3.1 provides the precise network architectures (described in Section 4.4 of the main paper) for the offense history (Table 2a of the main paper) and horizon (Table 2b of the main paper) experiments with our best method- CVAE with role alignment and conditionals consisting of 1s offense history, defense future context, and the team identity. The architectures for defense and mixed prediction are similar. All layers are fully connected with elu activation, except the last layer of every decoder, which has sigmoid activation. The player team encoder has layers with dimension 20-10-5. The player identity encoder has layers with dimension 500-250-100-64-32-16.

Learning At train time we minimize the loss via backpropagation with the ADAM optimizer, batch size 256, initial learning rate 0.001, and 0.5 learning rate decay every 10 epochs of size 200K. We also randomly sample the training set so that the number of times a sequence appears in an epoch is proportional to the number of players it has with unique identity.

3.5 Experiments

We evaluate the effect on prediction performance of: 1) each information type input in our proposed model architecture (Section 3.5); 2) the number and types of agents in the input and output, *i.e.*, offense only, defense only, and both offense and defense (Section 3.5); 3) the predicted agents' during-play role (Section 3.5); 4) the length of the history input (Section 3.5); and 5) the length of the prediction horizon (Section 3.5).

Baselines Our baselines are: velocity-based extrapolation, nearest neighbor retrieval, vanilla and Social LSTMs, and a VAE. Retrieval was performed using nearest neighbor search on the aligned (Section 3.4) trajectory history of the agents we wish to predict, matching the evaluation track histories to the training track histories based on minimum Euclidean distance. Then, we compare the error of the future trajectories of the top-k results to the ground truth. We found that these predictions are very poor, performing significantly worse than velocity-based extrapolation. Next, we compared our performance with the previous state of the art recurrent prediction methods, namely a vanilla LSTM and the Social LSTM. We found that the vanilla LSTM performed poorly with around 25 ft error for 4 s prediction horizon. The inclusion of social pooling improved the performance of the LSTM with 18 ft error for 4 s prediction horizon. However, the Social LSTM still performed significantly worse than simple velocity extrapolation at time horizons less than 6 s. The poor performances of the vanilla LSTM method and the Social LSTM method agrees with previous work on predicting basketball player trajectories conducted on a different data set [117]. As such, for most experiments, we use velocity-based extrapolation as our baseline, since it has the best performance.

Performance metrics We report three metrics. First, the L_2 distance (ft) between predicted trajectories and the ground truth, averaged over each time step for each agent. Second is the maximum distance between the prediction and ground truth for an agent trajectory, averaged over all agent trajectories. Last is the miss rate, calculated as the fraction of time the L_2 error exceeds 3 ft.

Method	**Alignment**	**Conditional**			**Error** (Offense, 4 s in future)		
		History	Context	Identity	Avg dist [ft] / (Top-5)	Max dist	Miss rate
Velocity	-	-	-	-	7.77	14.45	82.18
Retrieval	role	-	-	-	11.41 / (8.80)	28.57	86.77
VAE	random	-	-	-	7.10	19.24	74.90
VAE	role	-	-	-	6.85	18.84	72.78
CVAE	random	1 s	none	none	6.90	18.98	73.83
CVAE	random	none	encoded	none	6.97	18.46	75.29
CVAE	random	none	none	team	7.05	19.25	74.15
CVAE	random	none	none	player	7.02	19.17	75.15
CVAE	random	none	encoded	team	6.98	18.46	75.65
CVAE	random	1 s	none	team	6.91	18.95	74.18
CVAE	random	1 s	encoded	none	6.73	18.11	74.64
CVAE	random	1 s	encoded	team	6.76	18.15	74.97
CVAE	random	1 s	encoded	player	6.64	18.00	74.29
CVAE	position	1 s	encoded	team	6.09	16.87	70.37
CVAE	**role**	**1 s**	**encoded**	**none**	5.81	**16.41**	66.67
CVAE	**role**	**1 s**	**encoded**	**team**	**5.80**	16.45	**66.39**
CVAE	role	1 s	encoded	player	5.96	17.03	67.07

Table 3.2: **Offense prediction error for 4 s history and prediction horizon**. We test three different trajectory alignments i) random, ii) canonical position, and iii) role. We also test 3 conditionals: a) the previous one second of player motions (history), b) the next 4 s of the defensive motions (context), and c) one-hot encoded player or team (identity). The miss rate is calculated with threshold 3 feet.

What information gives us the best prediction?

In our proposed problem, there are four sources of information with the potential to improve prediction: i) the trajectory history $X_A^{[t_0,t_q]}$ of all agents, ii) the future motion $X_K^{(t_q,t_f]}$ of the players not predicted, *i.e.*, context, iii) the player/team identities, *i.e.*, personalization and iv) the agent alignment. The observed trajectory history serves as the input to the model and is fixed to 4 s. The final 1 second of trajectory history of the players we predict, the context, and the identity are treated as conditionals (Fig. 3.3), whereas the agent alignment enables efficient trajectory encoding. For this section (Table 3.2), we only predict the offense, which avoids conflating the effect of agent type with the effect of the information sources. We also fix the prediction horizon at 4 s.

To understand the influence of alignment alone, we compare the result of the baseline VAE with random versus role aligned agents. In the absence of the alignment the VAE has moderate performance, outperforming baselines. For example, in the first row of Fig. 3.4 the VAE captures

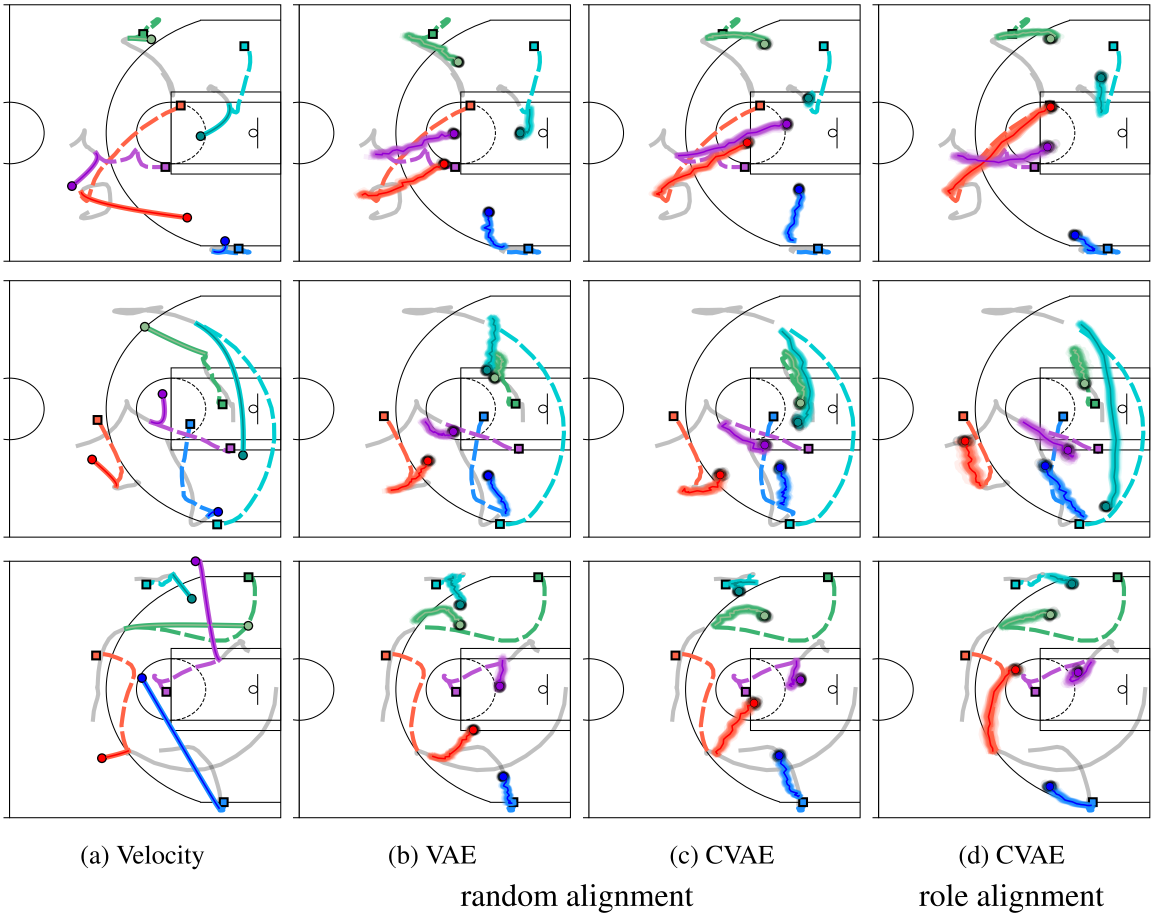


Figure 3.4: **Offense player predictions.** Given a 4 s trajectory history (gray) for all players (defense not pictured), we predict (solid lines) the next 4 s of offense player motion. Dashed lines are ground truth. Each row represents the same play, and each trajectory color corresponds to a player. The color intensity is proportional to the likelihood. **Column a)** velocity-based extrapolation. **Column b)** VAE with random trajectory alignment. **Column c)** CVAE with random trajectory alignment and all conditionals (player ID). **Column d)** adding role alignment to the CVAE (team ID).

co-movement of players (red and purple) that velocity-based extrapolation does not. However, the VAE does not capture the two agents crossing.

To understand the influence of each conditional, we randomly order the input trajectories and perform a set of ablation studies using a variety of conditions. We apply each conditional separately to compare their individual effects on performance, including comparing the use of team versus player identity.

Interestingly, the VAE and the CVAE using a single conditional perform similarly. However, if we combine conditionals, we create an even stronger co-movement signal, *e.g.*, red and purple players in the first row in Fig. 3.4. Still, with all the conditionals and random agent ordering, we

Method-Align-Pl.	Personnel	Error: Avg dist [ft]		
		1 s	4 s	8 s
Velocity	offense	7.74	7.72	7.74
CVAE-rand-ID	offense	7.06	**6.64**	6.86
CVAE-role-none	offense	6.04	**5.81**	6.21
CVAE-role-team	offense	6.05	**5.80**	6.16
CVAE-role-team	defense	4.23	**4.10**	4.31
CVAE-role-team	mix 5v5	5.75	5.74	5.76

(a) observed history

Method-Align-Pl.	Personnel	Error: Avg dist [ft] (4 s history)				
		1 s	2 s	4 s	6 s	8 s
Velocity	offense	**1.93**	4.10	7.72	11.50	24.02
CVAE-rand-ID	offense	2.66	4.23	6.64	8.14	9.41
CVAE-role-none	offense	2.38	4.00	5.81	7.07	8.28
CVAE-role-team	offense	2.35	**3.95**	**5.80**	**7.08**	**8.07**
CVAE-role-team	defense	2.08	3.01	4.10	4.98	5.85
Vanilla LSTM	mix 5v5	10.44	18.29	25.36	28.07	29.56
Social LSTM	mix 5v5	5.23	11.08	17.95	20.88	22.38
CVAE-role-team	mix 5v5	**2.44**	**3.92**	**5.74**	**7.21**	**8.33**

(b) prediction horizon

Method: CVAE-role-team

Mixture	Error: Avg dist [ft]
1v1	4.19
2v2	4.88
3v3	5.21
4v4	5.28
5v5	5.74

(c) num. players

Table 3.3: **Prediction error ablation.** a) We vary the observed history for a 4 s prediction, and observe that the optimal trajectory history is 4 s, though marginally so. b) We vary the prediction horizon given a 4 s observed history, and observe that the prediction error monotonically increases as a function of time horizon. c) We vary the number of players to predict for a 4 s horizon given a 4 s history, and observe an increase in average prediction error as we increase the number of agents per team from 1 to 5. For all experiments, we conditioned on the previous 1 s, the future motion of all agents not predicted, and the selected player or team identities. All errors are in feet.

fail to get the crossing of the trajectories.

When we both align and condition, we are able to correctly predict tracks crossing (red and purple players first row in Fig. 3.4d). In particular, we see the greatest improvement in our prediction by including the context, history, and team identity (bold in Table 3.2). These results imply that alignment, context, and history contain complementary information. Though alignment and conditioning improve our predictions, we struggle to predict sudden changes in movement (red player in row 3 of Fig. 3.4d), and stationary players (green players in row 1 and blue player in row 3 of Fig. 3.4d).

The modest improvements found by including team identity vanish when we use multi-template tree-based role alignment; implying that the alignment contains the added information provided by conditioning on the team identity. In other words, the clusters in latent space that the variational module finds with canonical alignment are team sensitive. This sensitivity to the team implies that certain teams perform certain collective motions. However, after tree-alignment, this vanishes, implying that the clusters found given optimal alignment exist below the level of player combinations.

How many and which agents can we predict?

To evaluate how many and which agents we can predict, we split our prediction tasks into i) exclusively predicting all 5 offense agents (Section 3.5), ii) exclusively predicting all 5 defense agents, and iii) predicting a mixture of offense and defense agents, from one of each (mix 1v1) to all 10 agents (mix 5v5).

Defense only Predicting defense is more straightforward than our other tasks because the defense reacts to the offense's play. Thus, the offense motion encodes much of the information about the defense motion. This is supported by the overall improvement in prediction for the defense as compared to the offense (Table 3.3a and b). The trends in the effect of conditionals and alignment are similar to the offense-only prediction results, indicating the value of information is similar regardless of adversary predicted. Therefore, we use role alignment and conditionals history, context, and team identity in subsequent experiments.

Mixed offense and defense Our most challenging prediction task is to simultaneous predict the motion of offense and defense. This is akin to asking: can we predict the motion of unobserved agents given the motion of the remaining seen agents? In the most general case of trying to predict all players, we found that the prediction performance splits the difference between the prediction of the offense and defense alone (Table 3.3a).

Next, we investigated how many agents per team we could predict over a 4 s time horizon, given a 4 s history (Table 3.3c). Surprisingly, we found relatively little performance degradation when predicting the motion of all ten players (5v5) versus one player each (1v1) on offense and defense (5.7 ft vs 4.2 ft). In the case of predicting all ten agents, the only conditionals are the player or team identities and the previous 1 s of history. The input is the 4 s trajectory history.

How does personnel influence prediction?

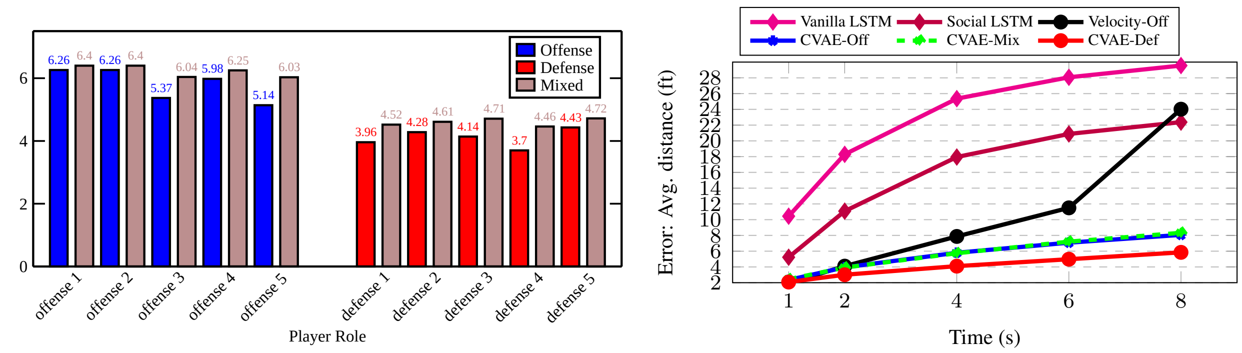
Since alignment improved our prediction results, we investigated the per-role prediction error (Fig. 3.5a) to uncover whether some roles are easier to predict than others. We found $\sim 16\%$ difference in the per-role prediction error for predicting offense compared to defense only. However, the per role variation does not hold when predicting a mixture of agents, in which case the prediction error of all agents increases.

How much history do we need?

Next, we tested the effect of the observed trajectory duration on prediction performance, that is how the history length influences predictions. The conditionals are the previous 1 s of the agents we are predicting, the future motion of players we are not predicting, and the team or player identity. We varied the observed history from 1-8 s and predicted the subsequent 4 s. As before, the defense is the easiest to predict, and multi-template role alignment with team identity provides the best prediction performance (Table 3.3a). We find 4 s of history is barely optimal, either because the player motions decorrelate at this time scale, or our encoder architecture cannot recover correlations at longer timescales.

How far can we predict?

To evaluate how far in the future we can predict, we provided 4 s of history of all player motions and predicted out to at most 8 s. Additionally, we provided the last 1 s of player motions and the future of the un-predicted agents as a conditional. In Fig. 3.6 we can clearly see that as the we



(a) **Per-role error (avg distance, ft).**

(b) **Error vs prediction horizon.**

Figure 3.5: **Prediction error ablation.** For all experiments, we provided 4 s of history and conditioned on the previous 1 second and the future of all agents not predicted. a) We evaluate the per-role prediction error for a 4 s prediction horizon, given a 4 s observed history. Defense is easier to predict than offense, and although mixed (2v2) appears to have better overall prediction than offense, per-role it is slightly worse, which makes sense because it's a harder task. b) We visualize the prediction errors as a function of horizon, given a 4 s observed trajectory history. The baselines are velocity (for offense only), and vanilla LSTM and Social LSTM (for all 10 agents), which we compare with our best method run on offense and defense only, as well as the mixture of all 10 agents. The precise values are reported in Table 3.3b.

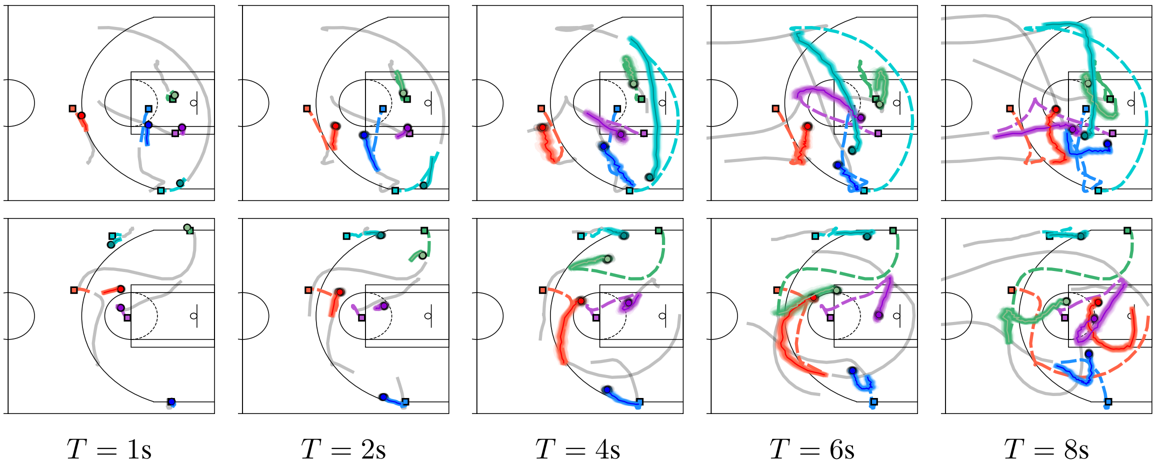


Figure 3.6: **Prediction as a function of time horizon.** We input the previous 4 s of every agent's motion (grey), and predict the offense player trajectories over horizon T s. The conditionals are the future motion of the defense (not shown), the final one second of offense history, and team identity. Each row represents a different example, and each color represents the player tree-based role. Dashed lines are the ground truth.

to underestimate the curvature of motions (cyan in example 1, $T = 6\ s$), or underestimate the complexity of motion (purple in row 1, $T = 6\ s$ and red in row 2, $T = 6\ s$).

As expected, the prediction error increases monotonically with the prediction time horizon (Fig. 3.5b), and when we include team identity, the prediction error changes less with the time horizon. Also, we see that the prediction error for the defensive is smaller than mixed offense and defense or offense alone.

We also notice that we far outperform the current state of the art prediction methods (Fig. 3.5b). It is remarkable that even when predicting the motion of all agents that our performance is three times as good as the Social LSTM (for 4 s time horizon). Again, it is important to note that the performance of the LSTM baselines agrees with previous results on a similar dataset [117]. Lastly, we note that the prediction of player trajectories presented by Shan et al. [117] which uses far more information, specifically the egocentric appearance of all players produces a per player average error of 11.8 ft (3.6 m). Though not directly comparable, this shows the power of our proposed generative method: with less information, our method produces noticeably better results.

3.6 Discussion

We have shown that a generative method based on conditional variational autoencoder (CVAE) is three times as accurate as the state of the art recurrent frameworks for the task of predicting player trajectories in an adversarial team game. Furthermore, these predictions improve by conditioning the predictions on the history and the context, *i.e.*, the motion of agents not predicted and their identity. Also, where available, further improvement in the quality of prediction can be found by providing multi-template aligned data. By aligning and conditioning of context and history, we can produce remarkably accurate, context-specific predictions without the need for ranking and refinement modules. We also found that our predictions were sensitive to the player role, as determined during alignment. However, we did not find any additional improvement in prediction when providing the player identity alone. The sensitivity to the player role, but not identity implies that role contains the information held in identity alone. Therefore, more fine-grained personalization may require additional player data, such as weight, height, age, minutes played.

Chapter 4

Learning 3D Human Dynamics

In the previous two chapters, we focused on methods for prediction largely in an abstracted, overhead representation of game play in sports. This representation captures well player spacing and broad motion. However, with it, we lose all information about appearance, including pose and the individual person dynamics, such as their gait. Furthermore, the overhead representation does not necessarily generalize well; we may not always be able to estimate an overhead representation of people interacting. This necessitates the ability to predict directly from the pixels, which in turn means we need to be able to understand how people are moving in videos from looking at their appearance. In this chapter, we present a framework for learning a representation of human motion that we can: 1) use to estimate the 3d pose and shape of people moving in videos, and 2) use to hallucinate the motion surrounding a single-frame snapshot.

4.1 Introduction

Consider the image of the baseball player mid-swing in Figure 4.1. Even though we only see a flat two-dimensional picture, we can infer the player's 3D pose, as we can easily imagine how his knees bend and arms extend in space. Furthermore, we can also infer his motion in the surrounding moments as he swings the bat through. We can do this because we have a mental model of 3D human dynamics that we have acquired from observing many examples of people in motion.

In this work, we present a computational framework that can similarly learn a model of 3D human dynamics from video. Given a temporal sequence of images, we first extract per-image features, and then train a simple 1D temporal encoder that learns a representation of 3D human dynamics over a temporal context of image features. We force this representation to capture 3D human dynamics by predicting not only the current 3D human pose and shape, but also changes in pose in the nearby past and future frames. We transfer the learned 3D dynamics knowledge to static images by learning a hallucinator that can hallucinate the temporal context representation

This chapter is based on joint work with Angjoo Kanazawa, Jason Y. Zhang, and Jitendra Malik [61], presented primarily as it appeared in the CVPR 2019 proceedings.

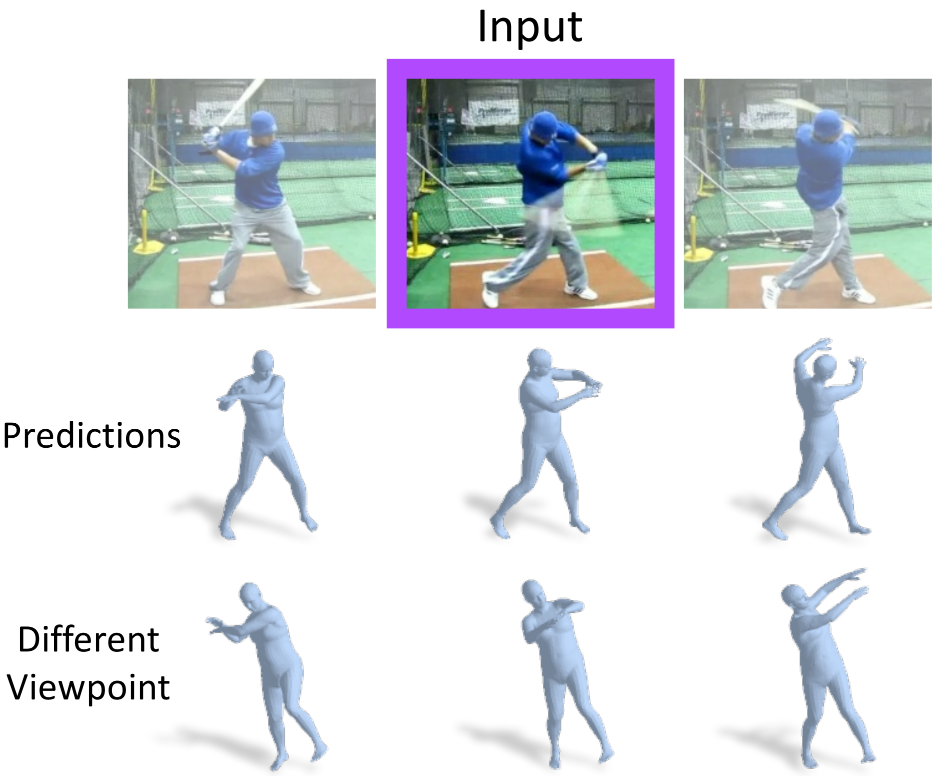


Figure 4.1: 3D motion prediction from a single image. We propose a method that, given a single image of a person, predicts the 3D mesh of the person's body and also hallucinates the future and past motion. Our method can learn from videos with only 2D pose annotations in a semi-supervised manner. Note our training set does not have any ground truth 3D pose sequences of batting motion. Our model also produces smooth 3D predictions from video input.

from a single image feature. The hallucinator is trained in a self-supervised manner using the actual output of the temporal encoder. Figure 4.2 illustrates the overview of our training procedure.

At test time, when the input is a video, the temporal encoder can be used to produce smooth 3D predictions: having a temporal context reduces uncertainty and jitter in the 3D prediction inherent in single-view approaches. The encoder provides the benefit of learned smoothing, which reduces the acceleration error by 56% versus a comparable single-view approach on a recent dataset of 3D humans in the wild. Our approach also obtains state-of-the-art 3D error on this dataset without any fine-tuning. When the input is a single image, the hallucinator can predict the current 3D human mesh as well as the change in 3D pose in nearby future and past frames, as illustrated in Figure 4.1.

We design our framework so that it can be trained on various types of supervision. A major challenge in 3D human prediction from a video or an image is that 3D supervision is limited in quantity and challenging to obtain at a large scale. Videos with 3D annotations are often captured in a controlled environment, and models trained on these videos alone do not generalize to the complexity of the real world. When 3D ground truth is not available, our model can be trained with 2D pose annotations via the reprojection loss [136] and an adversarial prior that constrains the 3D human pose to lie in the manifold of real human poses [60]. However, the amount of video labeled with ground truth 2D pose is still limited because ground truth annotations are costly to acquire.

While annotated data is always limited, there are millions of videos uploaded daily on the Internet. In this work we harvest this potentially unlimited source of unlabeled videos. We curate two large-scale video datasets of humans and train on this data using pseudo-ground truth 2D pose obtained from a state-of-the-art 2D pose detector [23]. Excitingly, our experiments indicate that adding more videos with pseudo-ground truth 2D monotonically improves the model performance both in term of 3D pose and 2D reprojection error: 3D pose error reduces by 9% and 2D pose accuracy increases by 8%. Our approach falls in the category of omni-supervision [103], a subset of semi-supervised learning where the learner exploits all data along with Internet-scale unlabeled data. We distill the knowledge of an accurate 2D pose detector into our 3D predictors through unlabeled video. While omni-supervision has been shown to improve 2D recognition problems, as far as we know, our experiment is the first to show that training on pseudo-ground truth 2D pose labels improves 3D prediction.

In summary, we propose a simple but effective temporal encoder that learns to capture 3D human dynamics. The learned representation allows smooth 3D mesh predictions from video in a feed-forward manner. The learned representation can be transferred to a static image, where from a single image, we can predict the current 3D mesh as well as the change in 3D pose in nearby frames. We further show that our model can leverage an Internet-scale source of unlabeled videos using pseudo-ground truth 2D pose.

4.2 Related Work

3D pose and shape from a single image. Estimating 3D body pose and shape from a single image is a fundamentally ambiguous task that most methods deal by using some model of human bodies and priors. Seminal works in this area [45, 111, 2] rely on silhouette features or manual interaction from users [111, 47, 148] to fit the parameters of a statistical body model. A fully automatic method was proposed by Bogo [16], which fits the parametric SMPL [83] model to 2D joint locations detected by an off-the-shelf 2D pose detector [102] with strong priors. Lassner [74] extend the approach to fitting predicted silhouettes. [144] explore the multi-person setting. Very recently, multiple approaches integrate the SMPL body model within a deep learning framework [120, 118, 99, 60, 96], where models are trained to directly infer the SMPL parameters. These methods vary in the cues they use to infer the 3D pose and shape: RGB image [118, 60], RGB image and 2D keypoints [120], keypoints and silhouettes [99], or keypoints and body part segmentations [96]. Methods that employ silhouettes obtain more accurate shapes, but require that the person is fully visible and unoccluded in the image. Varol explore predicting a voxel representation of human body [122]. In this work we go beyond these approaches by proposing a method that can predict shape and pose from a single image, as well as how the body changes locally in time.

3D pose and shape from video. While there are more papers that utilize video, most rely on a multi-view setup, which requires significant instrumentation. We focus on videos obtained from a monocular camera. Most approaches take a two-stage approach: first obtaining a single-view 3D reconstruction and then post-processing the result to be smooth via solving a constrained optimization problem [150, 131, 106, 108, 53, 91, 101]. Recent methods obtain accurate shapes and textures of clothing by pre-capturing the actors and making use of silhouettes [119, 137, 48, 8]. While these approaches obtain far more accurate shape, reliance on the pre-scan and silhouettes restricts these approaches to videos obtained in an interactive and controlled environments. Our approach is complementary to these two-stage approaches, since all predictions can be post-processed and refined. There are some recent works that output smooth 3D pose and shape: [120] predicts SMPL parameters from two video frames by using optical flow, silhouettes, and keypoints in a self-supervised manner. [7] exploits optical flow to obtain temporally coherent human poses. [59] fits a body model to a sequence of 3D point clouds and 3D joints obtained from multi-view stereo. Several approaches train LSTM models on various inputs such as image features [81], 2D joints [51], or 3D joints [29] to obtain temporally coherent 3D joint outputs. More recently, TP-Net [31] learns a fully convolutional network that smooths the predicted 3D joints. Concurrently to ours, [100] use a fully convolutional network to predict 3D joints from 2D joint sequences. We directly predict the 3D mesh outputs from 2D image sequences and can be trained with images without any ground truth 3D annotation. Furthermore, our temporal encoder predicts the 3D pose changes in nearby frames in addition to the current 3D pose. Our experiments indicate that the prediction losses help the encoder to pay more attention to the dynamics information available in the temporal window.

Learning motion dynamics. There are many methods that predict 2D future outputs from video using pixels [37, 33], flow [129], or 2D pose [130]. Other methods predict 3D future from 3D inputs [41, 57, 20, 80, 123]. In contrast, our work predicts future and past 3D pose from 2D inputs. There are several approaches that predict future from a single image [127, 138, 25, 79, 42], but all approaches predict future in 2D domains, while in this work we propose a framework that predicts 3D motions. Closest to our work is that of Chao [25], who forecast 2D pose and then estimate the 3D pose from the predicted 2D pose. In this work, we predict dynamics directly in the 3D space and learn the 3D dynamics from video.

4.3 Approach

Our goal is to learn a representation of 3D human dynamics from video, from which we can 1) obtain smooth 3D prediction and 2) hallucinate 3D motion from static images. In particular, we develop a framework that can learn 3D human dynamics from unlabeled, everyday videos of people on the Internet. We first define the problem and discuss different tiers of data sources our approach can learn from. We then present our framework that learns to encode 3D human motion dynamics from videos. Finally, we discuss how to transfer this knowledge to static images such that one can hallucinate short-term human dynamics from a static image. Figure 4.2 illustrates the framework.

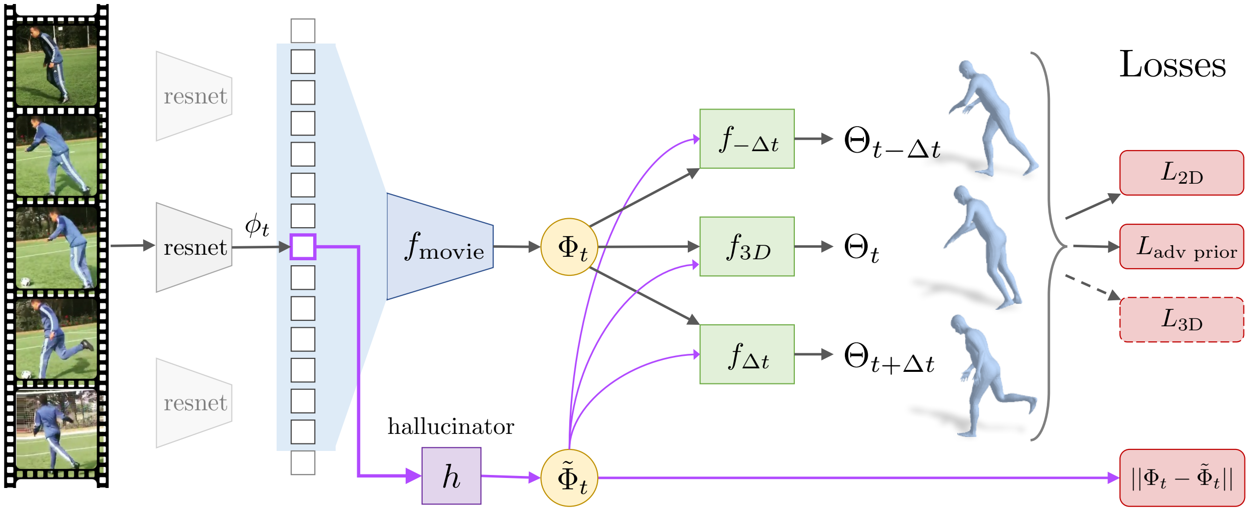


Figure 4.2: Overview of the proposed framework. Given a temporal sequence of images, we first extract per-image features ϕ_t. We train a temporal encoder f_{movie} that learns a representation of 3D human dynamics Φ_t over the temporal window centered at frame t, illustrated in the blue region. From Φ_t, we predict the 3D human pose and shape Θ_t, as well as the change in pose in the nearby $\pm\Delta t$ frames. The primary loss is 2D reprojection error, with an adversarial prior to make sure that the recovered poses are valid. We incorporate 3D losses when 3D annotations are available. We also train a hallucinator h that takes a single image feature ϕ_t and learns to hallucinate its temporal representation $\tilde{\Phi}_t$. At test time, the hallucinator can be used to predict dynamics from a single image.

Problem Setup

Our input is a video $V = \{I_t\}_{t=1}^T$ of length T, where each frame is a bounding-box crop centered around a detected person. We encode the tth image frame I_t with a visual feature ϕ_t, obtained from a pretrained feature extractor. We train a function f_{movie} that learns a representation Φ_t that encodes the 3D dynamics of a human body given a temporal context of image features centered at frame t. Intuitively, Φ_t is the representation of a "movie strip" of 3D human body in motion at frame t. We also learn a hallucinator $h : \phi_t \mapsto \Phi_t$, whose goal is to hallucinate the movie strip representation from a static image feature ϕ_t.

We ensure that the movie strip representation Φ_t captures the 3D human body dynamics by predicting the 3D mesh of a human body from Φ_t at different time steps. The 3D mesh of a human body in an image is represented by 85 parameters, denoted by $\Theta = \{\boldsymbol{\beta}, \boldsymbol{\theta}, \Pi\}$, which consists of shape, pose, and camera parameters. We use the SMPL body model [83], which is a function $\mathcal{M}(\boldsymbol{\beta}, \boldsymbol{\theta}) \in \mathbb{R}^{N\times 3}$ that outputs the $N = 6890$ vertices of a triangular mesh given the shape $\boldsymbol{\beta}$ and pose $\boldsymbol{\theta}$. Shape parameters $\boldsymbol{\beta} \in \mathbb{R}^{10}$ define the linear coefficients of a low-dimensional statistical shape model, and pose parameters $\boldsymbol{\theta} \in \mathbb{R}^{72}$ define the global rotation of the body and the 3D relative rotations of the kinematic skeleton of 23 joints in axis-angle representation. Please see [83] for more details. The mesh vertices define 3D locations of k joints $X \in \mathbb{R}^{k\times 3} = W\mathcal{M}(\boldsymbol{\beta}, \boldsymbol{\theta})$ via a pre-trained linear regressor $W \in \mathbb{R}^{k\times N}$. We also solve for the weak-perspective camera

$\Pi = [s, t_x, t_y]$ that projects the body into the image plane. We denote $x = \Pi(X(\boldsymbol{\beta}, \boldsymbol{\theta}))$ as the projection of the 3D joints.

While this is a well-formed supervised learning task if the ground truth values were available for every video, such 3D supervision is costly to obtain and not available in general. Acquiring 3D supervision requires extensive instrumentation such as a motion capture (MoCap) rig, and these videos captured in a controlled environment do not reflect the complexity of the real world. While more practical solutions are being introduced [88], 3D supervision is not available for millions of videos that are being uploaded daily on the Internet. In this work, we wish to harness this potentially infinite data source of unlabeled video and propose a framework that can learn 3D motion from pseudo-ground truth 2D pose predictions obtained from an off-the-shelf 2D pose detector. Our approach can learn from three tiers of data sources at once: First, we use the MoCap datasets $\{(V_i, \boldsymbol{\Theta}_i, x_i)\}$ with full 3D supervision $\boldsymbol{\Theta}_i$ for each video along with ground truth 2D pose annotations for k joints $x_i = \{x_t \in \mathbb{R}^{k\times 2}\}_{t=1}^{T}$ in each frame. Second, we use datasets of videos in the wild obtained from a monocular camera with human-annotated 2D pose: $\{(V_i, x_i)\}$. Third, we also experiment with videos with *pseudo*-ground truth 2D pose: $\{(V_i, \tilde{x}_i)\}$. See Table 4.1 for the list of datasets and their details.

Learning 3D Human Dynamics from Video

A dynamics model of a 3D human body captures how the body changes in 3D over a small change in time. Therefore, we formulate this problem as learning a temporal representation that can simultaneously predict the current 3D body and pose changes in a short time period. To do this, we learn a temporal encoder f_{movie} and a 3D regressor f_{3D} that predict the 3D human mesh representation at the current frame, as well as delta 3D regressors $f_{\Delta t}$ that predict how the 3D pose changes in $\pm\Delta t$ time steps.

Temporal Encoder Our temporal encoder consists of several layers of a 1D fully convolutional network f_{movie} that encodes a temporal window of image features centered at t into a representation Φ_t that encapsulates the 3D dynamics. We use a fully convolutional model for its simplicity. Recent literature also suggests that feed-forward convolutional models empirically out-perform recurrent models while being parallelizable and easier to train with more stable gradients [12, 92]. Our temporal convolution network has a ResNet [50] based architecture similar to [12, 1].

The output of the temporal convolution network is sent to a 3D regressor $f_{\text{3D}} : \Phi_t \mapsto \boldsymbol{\Theta}_t$ that predicts the 3D human mesh representation at frame t. We use the same iterative 3D regressor architecture proposed in [60]. Simply having a temporal context reduces ambiguity in 3D pose, shape, and viewpoint, resulting in a temporally smooth 3D mesh reconstruction. In order to train these modules from 2D pose annotations, we employ the reprojection loss [136] and the adversarial prior proposed in [60] to constrain the output pose to lie in the space of possible human poses. The 3D losses are also used when 3D ground truth is available. Specifically, the loss for the current frame consists of the reprojection loss on visible keypoints $L_{\text{2D}} = ||v_t(x_t - \hat{x}_t)||_2^2$, where $v_t \in \mathbb{R}^{k\times 2}$ is the visibility indicator over each keypoint, the 3D loss if available, $L_{\text{3D}} = ||\Theta_t - \hat{\Theta}_t||_2^2$, and the factorized adversarial prior of [60], which trains a discriminator D_k for each joint rotation of the body model $L_{\text{adv prior}} = \sum_k (D_k(\Theta) - 1)^2$. In this work, we regularize the shape predictions using a shape prior $L_{\beta \text{ prior}}$ [16]. Together the loss for frame t consists of $L_t = L_{\text{2D}} + L_{\text{3D}} + L_{\text{adv prior}} + L_{\beta \text{ prior}}$. Furthermore, each sequence is of the same person, so while the pose and camera may change every frame, the shape remains constant. We express this constraint as a constant shape loss over each sequence:

$$L_{\text{const shape}} = \sum_{t=1}^{T-1} ||\beta_t - \beta_{t+1}||. \tag{4.1}$$

Predicting Dynamics We enforce that the learned temporal representation captures the 3D human dynamics by predicting the 3D pose changes in a local time step $\pm\Delta t$. Since we are training with videos, we readily have the 2D and/or 3D targets at nearby frames of t to train the dynamics predictors. Learning to predict 3D changes encourages the network to pay more attention to the temporal cues, and our experiments show that adding this auxiliary loss improves the 3D prediction results. Specifically, given a movie strip representation of the temporal context at frame Φ_t, our goal is to learn a dynamics predictor $f_{\Delta t}$ that predicts the change in 3D parameters of the human body at time $t \pm \Delta t$.

In predicting dynamics, we only estimate the change in 3D pose parameters θ, as the shape should remain constant and the weak-perspective camera accounts for where the human is in the detected bounding box. In particular, to improve the robustness of the current pose estimation during training, we augment the image frames with random jitters in scale and translation which emulates the noise in real human detectors. However, such noise should not be modeled by the dynamics predictor.

For this task, we propose a dynamics predictor $f_{\Delta t}$ that outputs the 72D change in 3D pose $\Delta\theta$. $f_{\Delta t}$ is a function that maps Φ_t and the predicted current pose θ_t to the predicted change in pose $\Delta\theta$ for a specific time step Δt. The delta predictors are trained such that the predicted pose in the new timestep $\theta_{t+\Delta t} = \theta_t + \Delta\theta$ minimizes the reprojection, 3D, and the adversarial prior losses at time frame $t + \Delta t$. We use the shape predicted in the current time t to obtain the mesh for $t \pm \Delta t$ frames. To compute the reprojection loss without predicted camera, we solve for the optimal scale s and translation $\vec{t}$ that aligns the orthographically projected 3D joints $x_{\text{orth}} = X[:, : 2]$ with the visible ground truth 2D joints x_{gt}: $\min_{s,\vec{t}} ||(s x_{\text{orth}} + \vec{t}) - x_{gt}||_2$. A closed form solution exists for this problem, and we use the optimal camera $\Pi^* = [s^*, \vec{t}^*]$ to compute the reprojection error on poses predicted at times $t \pm \Delta t$. Our formulation factors away axes of variation, such as shape and camera, so that the delta predictor focuses on learning the temporal evolution of 3D pose. In summary, the overall objective for the temporal encoder is

$$L_{\text{temporal}} = \sum_t L_t + \sum_{\Delta t} L_{t+\Delta t} + L_{\text{const shape}}. \tag{4.2}$$

In this work we experiment with two Δt at $\{-5, 5\}$ frames, which amounts to ±0.2 seconds for a 25 fps video.

Hallucinating Motion from Static Images

Given the framework for learning a representation for 3D human dynamics, we now describe how to transfer this knowledge to static images. The idea is to learn a hallucinator $h : \phi_t \mapsto \tilde{\Phi}_t$ that maps a single-frame representation ϕ_t to its "movie strip" representation $\tilde{\Phi}_t$. One advantage of working with videos is that during training, the target representation Φ_t is readily available for every frame t from the temporal encoder. Thus, the hallucinator can be trained in a weakly-supervised manner, minimizing the difference between the hallucinated movie strip and the actual movie strip obtained from f_{movie}:

$$L_{\text{hal}} = ||\Phi_t - \tilde{\Phi}_t||_2. \tag{4.3}$$

Furthermore, we pass the hallucinated movie strip to the f_{3D} regressor to minimize the single-view loss as well as the delta predictors $f_{\Delta t}$. This ensures that the hallucinated features are not only similar to the actual movie strip but can also predict dynamics. All predictor weights are shared among the actual and hallucinated representations.

In summary we jointly train the temporal encoder, hallucinator, and the delta 3D predictors together with overall objective:

$$L = L_{\text{temporal}} + L_{\text{hal}} + L_t(\tilde{\Phi}_t) + \sum_{\Delta t} L_{t+\Delta t}(\tilde{\Phi}_t). \tag{4.4}$$

See Figure 4.2 for the overview of our framework.

4.4 Learning from Unlabeled Video

Although our approach can be trained on 2D pose annotations, annotated data is always limited – the annotation effort for labeling keypoints in videos is substantial. However, millions of videos are uploaded to the Internet every day. On YouTube alone, 300 hours of video are uploaded every minute [10].

Therefore, we curate two Internet-scraped datasets with pseudo-ground truth 2D pose obtained by running OpenPose [23]. An added advantage of OpenPose is that it detects toe points, which are not labeled in any of the video datasets with 2D ground truth. Our first dataset is VLOG-people, a subset of the VLOG lifestyle dataset [38] on which OpenPose fires consistently. To get a more diverse range of human dynamics, we collect another dataset, InstaVariety, from Instagram using 84 hashtags such as *#instruction*, *#swimming*, and *#dancing*. A large proportion of the videos we collected contain only one or two people moving with much of their bodies visible, so OpenPose [22] produced reasonably good quality 2D annotations. For videos that contain multiple people, we form our pseudo-ground truth by linking the per-frame skeletons from OpenPose using the Hungarian algorithm-based tracker from Detect and Track [43]. A clear advantage of unlabeled videos is that they can be easily collected at a significantly larger scale than videos with human-annotated 2D pose. Altogether, our pseudo-ground truth data has over 28 hours of 2D-annotated footage, compared to the 79 minutes of footage in the human-labeled datasets. See Table 4.1 for the full dataset comparison.

				Annotation Type		
Dataset Name	Total Frames	Total Length (min)	Avg. Length (sec)	GT 3D	GT 2D	In-the-wild
Human3.6M	581k	387	48	✓	✓	
Penn Action	77k	51	3		✓	✓
NBA (Ours)	43k	28	3		✓	✓
VLOG people (Ours)	353k	236 (4 hr)	8			✓
InstaVariety (ours)	**2.1M**	**1459** (1 day)	6			✓

Table 4.1: Three tiers of video datasets. We jointly train on videos with: full ground truth 2D and 3D pose supervision, only ground truth 2D supervision, and pseudo-ground truth 2D supervision. Note the difference in scale for pseudo-ground truth datasets.

4.5 Experimental Setup

Architecture: We use Resnet-50 [50] pretrained on single-view 3D human pose and shape prediction [60] as our feature extractor, where $\phi_i \in \mathbb{R}^{2048}$ is the the average pooled features of the last layer. Since training on video requires a large amount of memory, we precompute the image features on each frame similarly to [1]. This allow us to train on 20 frames of video with mini-batch size of 8 on a single 1080ti GPU. Our temporal encoder consists of 1D temporal convolutional layers, where each layer is a residual block of two 1D convolutional layers of kernel width of 3 with group norm. We use three of these layers, producing an effective receptive field size of 13 frames. The final output of the temporal encoder has the same feature dimension as ϕ. Our hallucinator contains two fully-connected layers of size 2048 with skip connection. Please see the supplementary material for more details.

Datasets: Human3.6M [55] is the only dataset with ground truth 3D annotations that we train on. It consists of motion capture sequences of actors performing tasks in a controlled lab environment. We follow the standard protocol [60] and train on 4 subjects (S1, S6, S7, S8) and test on 2 subjects (S9, S11) with 1 subject (S5) as the validation set.

For in-the-wild video datasets with 2D ground truth pose annotations, we use the Penn Action [146] dataset and our own NBA dataset. Penn Action consists of 15 sports actions, with 1257 training videos and 1068 test. We set aside 10% of the test set as validation. The NBA dataset contains videos of basketball players attempting 3-point shots in 16 basketball games. Each sequence contains one set of 2D annotations for a single player. We split the dataset into 562 training videos, 64 validation, and 151 test. Finally, we also experiment with the new pseudo-ground truth 2D datasets (Section 4.4). See Table 4.1 for the summary of each dataset. Unless otherwise indicated, all models are trained with Human3.6M, Penn Action, and NBA.

We evaluate our approach on the recent 3D Poses in the Wild dataset (3DPW) [88], which contains 61 sequences (25 train, 25 test, 12 val) of indoor and outdoor activities. Portable IMUs provide ground truth 3D annotations on challenging in-the-wild videos. To remain comparable to existing methods, we do not train on 3DPW and only used it as a test set. For evaluations on all datasets, we skip frames that have fewer than 6 visible keypoints.

As our goal is not human detection, we assume a temporal tube of human detections is avail-

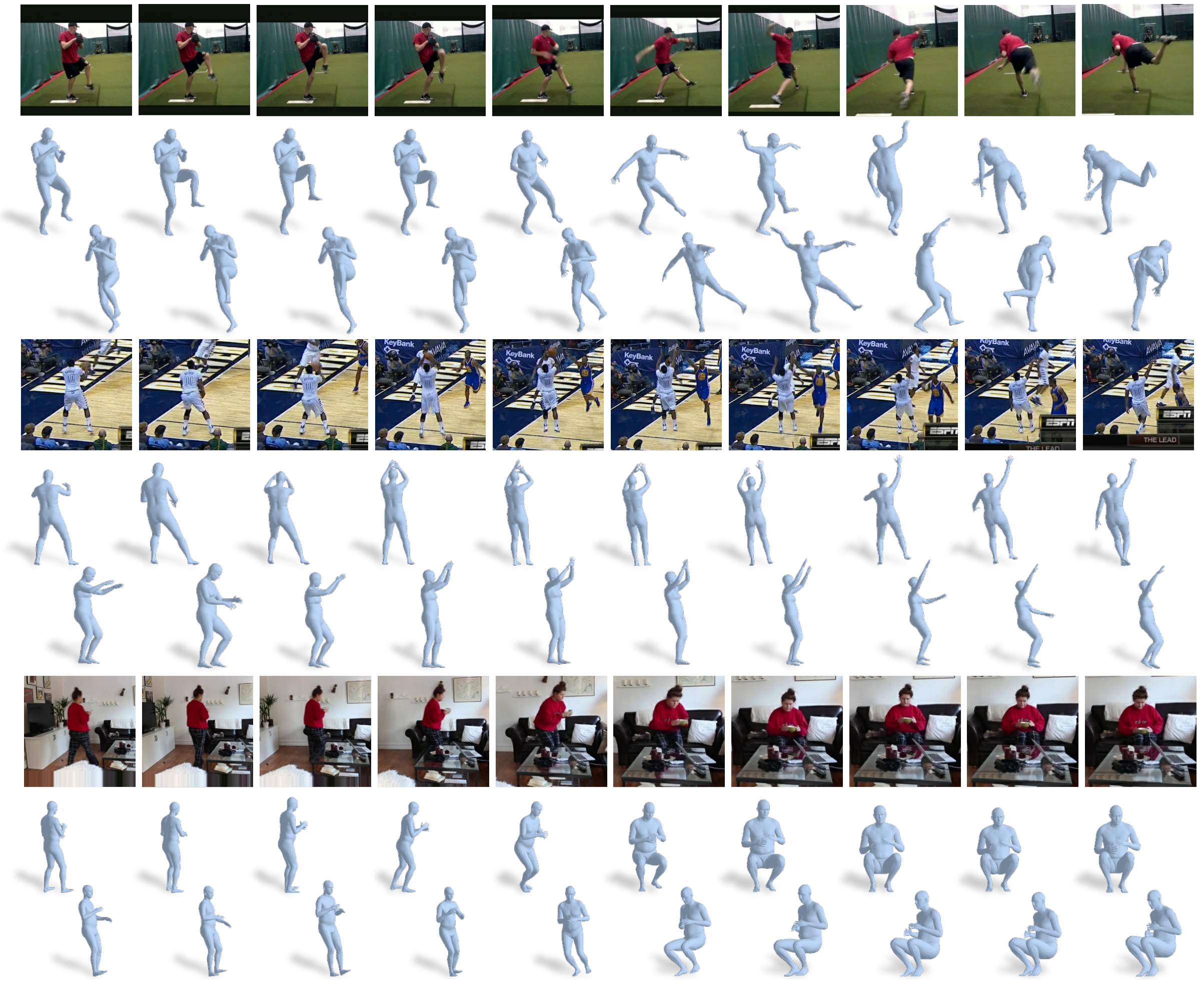


Figure 4.3: Qualitative results of our approach on sequences from Penn Action, NBA, and VLOG. For each sequence, the top row shows the cropped input images, the middle row shows the predicted mesh, and the bottom row shows a different angle of the predicted mesh. Our method produces smooth, temporally consistent predictions.

able. We use ground truth 2D bounding boxes if available, and otherwise use the output of OpenPose to obtain a temporally smooth tube of human detections. All images are scaled to 224x224 where the humans are roughly scaled to be 150px in height.

4.6 Experiments

We first evaluate the efficacy of the learned temporal representation and compare the model to local approaches that only use a single image. We also compare our approaches to state-of-the-art 3D pose methods on 3DPW. We then evaluate the effectiveness of training on pseudo-ground truth 2D poses. Finally, we quantitatively evaluate the dynamics prediction from a static

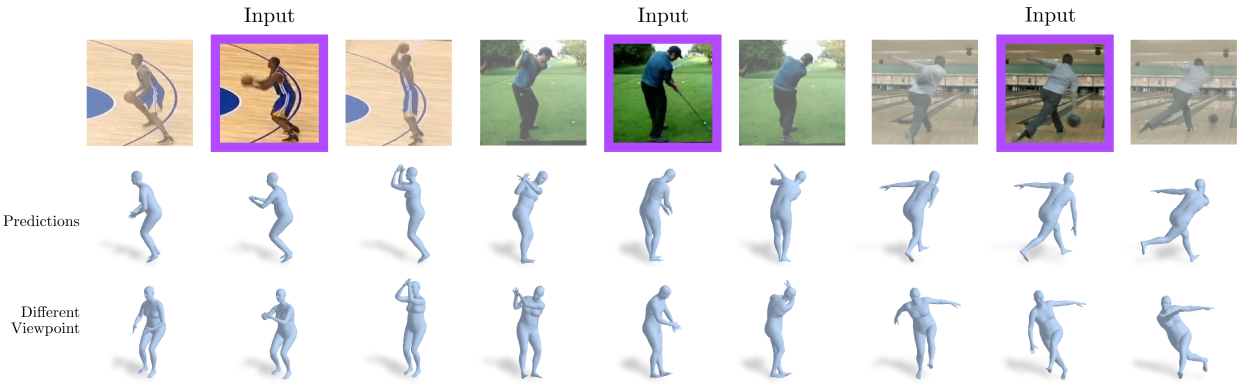


Figure 4.4: Predicting 3D dynamics. In the top row, the boxed image is the single-frame input to the hallucinator while the left and right images are the ground truth past and future respectively. The second and third rows show two views of the predicted meshes for the past, present, and future given the input image.

image on Human3.6M. We show qualitative results on video prediction in Figure 4.3 and static image dynamics prediction in Figure 4.1 and 4.4. Please see the supplementary for more ablations, metrics, and discussion of failure modes. In addition, a video with more of our results is available at `https://youtu.be/9fNKSZdsAG8`.

Local vs Temporal Context

	3DPW				NBA		Penn Action	
	PCK ↑	MPJPE ↓	PA-MPJPE ↓	Accel Error ↓	PCK ↑	Accel	PCK ↑	Accel
Single-view retrained [60]	84.1	130.0	**76.7**	37.4	55.9	163.6	73.2	79.9
Context. no dynamics	82.6	139.2	78.4	**15.2**	64.2	46.6	71.2	29.3
Contextual	**86.4**	**127.1**	80.1	16.4	**68.4**	44.1	**77.9**	29.7

Table 4.2: Local vs temporal context. Our temporal encoder produces smoother predictions, significantly lowering the acceleration error. We also find that training for dynamic prediction considerably improves 2D keypoint estimation.

We first evaluate the proposed temporal encoder by comparing with a single-view approach that only sees a local window of one frame. As the baseline for the local window, we use a model similar to [60], re-trained on the same training data for a fair comparison. We also run an ablation by training our model with our temporal encoder but without the dynamics predictions $f_{\Delta t}$.

In order to measure smooth predictions, we propose an *acceleration error*, which measures the average difference between ground truth 3D acceleration and predicted 3D acceleration of each joint in mm/s^2. This can be computed on 3DPW where ground truth 3D joints are available. On 2D datasets, we simply report the acceleration in mm/s^2.

	3DPW		H36M
	MPJPE ↓	PA-MPJPE ↓	PA-MPJPE ↓
Martinez [89]	-	157.0	47.7
SMPLify [16]	199.2	106.1	82.3
TP-Net [30]	163.7	92.3	**36.3**
Ours	**127.1**	**80.1**	58.1
Ours + InstaVariety	**116.5**	**72.6**	56.9

Table 4.3: Comparison to state-of-the-art 3D pose reconstruction approaches. Our approach achieves state-of-the-art performance on 3DPW. Good performance on Human3.6M does not always translate to good 3D pose prediction on in-the-wild videos.

We also report other standard metrics. For 3DPW, we report the mean per joint position error (MPJPE) and the MPJPE after Procrustes Alignment (PA-MPJPE). Both are measured in millimeters. On datasets with only 2D ground truth, we report accuracy in 2D pose via percentage of correct keypoints [140] with $\alpha = 0.05$.

We report the results on three datasets in Table 4.2. Overall, we find that our method produces modest gains in 3D pose estimation, large gains in 2D, and a very significant improvement in acceleration error. The temporal context helps to resolve ambiguities, producing smoother, temporally consistent results. Our ablation study shows that access to temporal context alone is not enough; using the auxiliary dynamics loss is important to force the network to learn the *dynamics* of the human.

Comparison to state-of-the-art approaches. In Table 4.3, we compare our approach to other state-of-the-art methods. None of the approaches train on 3DPW. Note that Martinez [89] performs well on the Human3.6M benchmark but achieves the worst performance on 3DPW, showing that methods trained exclusively on Human3.6M do not generalize to in-the-wild images. We also compare our approach to TP-Net, a recently-proposed semi-supervised approach that is trained on Human3.6M and MPII 2D pose in-the-wild dataset [9]. TP-Net also learns a temporal smoothing network supervised on Human3.6M. While this approach is highly competitive on Human3.6M, our approach significantly out-performs TP-Net on in-the-wild video. We only compare feed-forward approaches and not methods that smooth the 3D predictions via post-optimization. Such post-processing methods are complementary to feed-forward approaches and would benefit any of the approaches.

Training on pseudo-ground truth 2D pose

Here we report results of models trained on the two Internet-scale datasets we collected with pseudo-ground truth 2D pose annotations (See Table 4.4). We find that the adding more data

	3DPW			NBA	Penn
	PCK ↑	MPJPE ↓	PA-MPJPE ↓	PCK ↑	PCK ↑
Ours	86.4	127.1	80.1	**68.4**	77.9
Ours + VLOG	91.7	126.7	77.7	68.2	78.6
Ours + InstaVariety	**92.9**	**116.5**	**72.6**	68.1	**78.7**

Table 4.4: Learning from unlabeled video via pseudo ground truth 2D pose. We collected our own 2D pose datasets by running OpenPose on unlabeled video. Training with these pseudo-ground truth datasets induces significant improvements across the board.

	Past	Current	Future
	PA-MPJPE ↓	PA-MPJPE ↓	PA-MPJPE ↓
N.N.	71.6	**50.9**	70.7
Const.	68.6	58.1	69.3
Ours 1	**65.0**	58.1	**65.3**
Ours 2	65.7	60.7	66.3

Table 4.5: Evaluation of dynamic prediction on Human3.6M. The Nearest Neighbors baseline uses the pose in the training set with the lowest PA-MPJPE with the ground truth current pose to make past and future predictions. The constant baseline uses the current prediction as the future and past predictions. Ours 1 is the prediction model with Eq. 4.3, Ours 2 is that without Eq. 4.3.

monotonically improves the model performance both in terms of 3D pose and 2D pose reprojection error. Using the largest dataset, InstaVariety, 3D pose error reduces by 9% and 2D pose accuracy increases by 8% on 3DPW. We see a small improvement or no change on 2D datasets. It is encouraging to see that not just 2D but also 3D pose improves from pseudo-groundtruth 2D pose annotations.

Predicting dynamics

We quantitatively evaluate our static image to 3D dynamics prediction. Since there are no other methods that predict 3D poses from 2D images, we propose two baselines: a constant baseline that outputs the current frame prediction for both past and future, and an Oracle Nearest Neighbors baseline. We evaluate our method on Human3.6M and compare with both baselines in Table 4.5.

Clearly, predicting dynamics from a static image is a challenging task due to inherent ambiguities in pose and the stochasticity of motion. Our approach works well for ballistic motions in which there is no ambiguity in the direction of the motion. When it's not clear if the person is going up or down our model learns to predict no change.

4.7 Discussion

We propose an end-to-end model that learns a model of 3D human dynamics that can 1) obtain smooth 3D prediction from video and 2) hallucinate 3D dynamics on single images at test time. We train a simple but effective temporal encoder from which the current 3D human body as well as how the 3D pose changes can be estimated. Our approach can be trained on videos with 2D pose annotations in a semi-supervised manner, and we show empirically that our model can improve from training on an Internet-scale dataset with pseudo-groundtruth 2D poses. While we show promising results, much more remains to be done in recovering 3D human body from video. Upcoming challenges include dealing with occlusions and interactions between multiple people.

4.8 Supplementary

Model architecture

Temporal Encoder Figure 4.5 visualizes the architecture of our temporal encoder f_{movie}. Each 1D convolution has temporal kernel size 3 and filter size 2048. For group norm, we use 32 groups each with 64 channels. We repeat the residual block 3 times, which gives us a field of view of 13 frames.

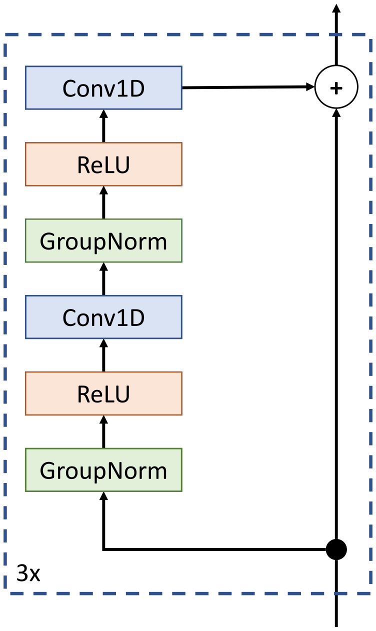


Figure 4.5: Architecture of the temporal encoder f_{movie}.

Hallucinator Our hallucinator consists of two fully-connected layers of 2048 neurons, whose output gets added to the original ϕ as a skip connection.

3D regressors Our f_{3D} regresses the 85D Θ_t vector in an iterative error feedback (IEF) loop [24, 60], where the current estimates are progressively updated by the regressor. Specifically, the regressor takes in the current image feature ϕ_t and current parameter estimate $\Theta_t^{(j)}$, and outputs corrections $\Delta\Theta_t^{(j)}$. The current estimate gets updated by this correction $\Theta_t^{(j+1)} = \Delta\Theta_t^{(j)} + \Theta_t^{(j)}$. This loop is repeated 3 times. We initialize the $\Theta_t^{(0)}$ to be the mean values $\bar{\Theta}$, which we also update as a learned parameter.

The regressor consists of two fully-connected layers, both with 1024 neurons, with a dropout layer in between, followed by a final layer that outputs the 85D outputs. All weights are shared.

The dynamics predictors $f_{\pm\Delta t}$ has a similar form, except it only outputs the 72-D changes in pose θ, and the initial estimate is set to the prediction of the current frame t, *i.e.*$\theta_{t+\Delta t}^{(0)} = \theta_t$. Each $f_{\pm\Delta t}$ learns a separate set of weights.

Additional Ablations and Evaluations

In Table 4.6, we evaluate our method and comparable methods on 2D/3D pose and 3D shape recovery. We provide another ablation of our approach where the constant shape loss (Eq. 1) is not used (Ours – Const). In addition, we include full results from our ablation studies.

Shape Evaluation To measure shape predictions, we report *Posed Mesh Error* (Mesh Pos), which computes the mean Euclidean distance between the predicted and ground truth 3D meshes. Since this metric is affected by the quality of the pose predictions, we also report *Unposed Mesh Error* (Mesh Unp), which computes the same but with a fixed T-pose to evaluate shape independently of pose accuracy. Both metrics are in units of *mm*. Note that accurately capturing the shape of the subject is challenging since only 4 ground truth shapes are available in Human3.6M when training.

Failure Modes

While our experiments show promising results, there is still room for improvement.

Smoothing Overall, our method obtains smooth results, but it can struggle in challenging situations, such as person-to-person occlusions or fast motions. Additionally, extreme or rare poses (*e.g.*stretching, ballet) are difficult to capture.

Dynamics Prediction Clearly, predicting the past and future dynamics from a single image is a challenging problem. Even for us humans, from a single image alone, many motions are ambiguous. Figure 4.6 visualizes a canonical example of such ambiguity, where it is unclear from the input center image if she is about to raise her arms or lower them. In these cases, our model learns to predict constant pose.

Furthermore, even the pose in a single image can be ambiguous, for example due to motion blur in videos. Figure 4.7 illustrates a typical example, where the tennis player's arm has disappeared and therefore the model cannot discern whether the person is facing left or right. When the current

	3DPW						H3.6M			Penn Action
	PCK ↑	MPJPE ↓	PA-MPJPE ↓	Accel Err ↓	Mesh Pos ↓	Mesh Unp ↓	MPJPE ↓	PA-MPJPE ↓	Accel Err ↓	PCK ↑
Martinez [89]	-	-	157.0	-	-	-	62.9	47.7	-	-
SMPLify [16]	-	199.2	106.1	-	211.2	61.2	-	82.3	-	-
TP-Net [31]	-	163.7	92.3	-	-	-	**52.1**	**36.3**	-	-
Ours	86.4	127.1	80.1	16.4	144.4	25.8	87.0	58.1	9.3	77.9
Ours + VLOG	91.7	126.7	77.7	15.7	147.4	29.7	85.9	58.3	9.3	78.6
Ours + InstaVariety	**92.9**	**116.5**	**72.6**	**14.3**	**138.6**	26.7	83.7	56.9	9.3	**78.7**
Single-view retrained [60]	84.1	130.0	76.7	37.4	144.9	**24.4**	94.0	59.3	23.9	73.2
Ours – Dynamics	82.6	139.2	78.4	15.2	155.2	24.8	88.6	58.3	**9.1**	71.2
Ours – Const	86.5	128.3	78.2	16.6	145.9	27.5	83.5	57.8	9.3	78.1

Table 4.6: Evaluation of baselines, ablations, and our proposed method on 2D and 3D keypoints and 3D mesh. We compare with three other feed-forward methods that predict 3D joints. None of the models are trained on 3DPW, all of the models are trained on H3.6M, and only our models are trained on Penn Action (TP-Net also uses MPII 2D dataset). We show that training with pseudo-ground truth 2D annotations significantly improves 2D and 3D predictions on the in-the-wild video dataset 3DPW. Single-view is retrained on our data. Ours – Dynamics is trained without the past and future regressors $f_{\pm\Delta t}$. Ours – Const is trained without $L_{\text{const shape}}$.

frame prediction is poor, the resulting dynamics predictions are also not correct, since the dynamics predictions are initialized from the pose of the current frame.

Note that incorporating temporal context resolves many of these static-image ambiguities.

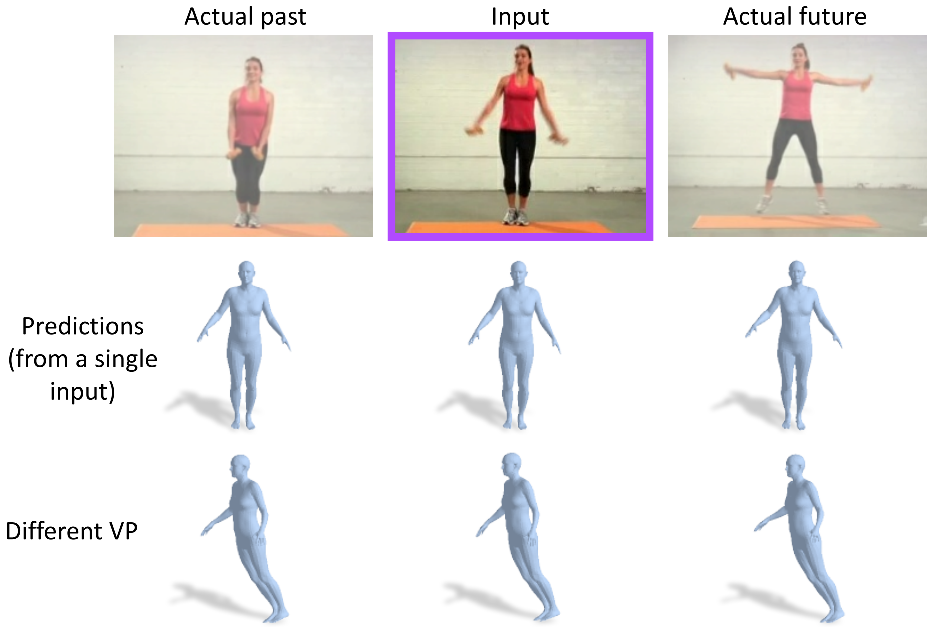


Figure 4.6: Ambiguous motion. Dynamic prediction is difficult from the center image alone, where her arms may reasonably lift or lower in the future.

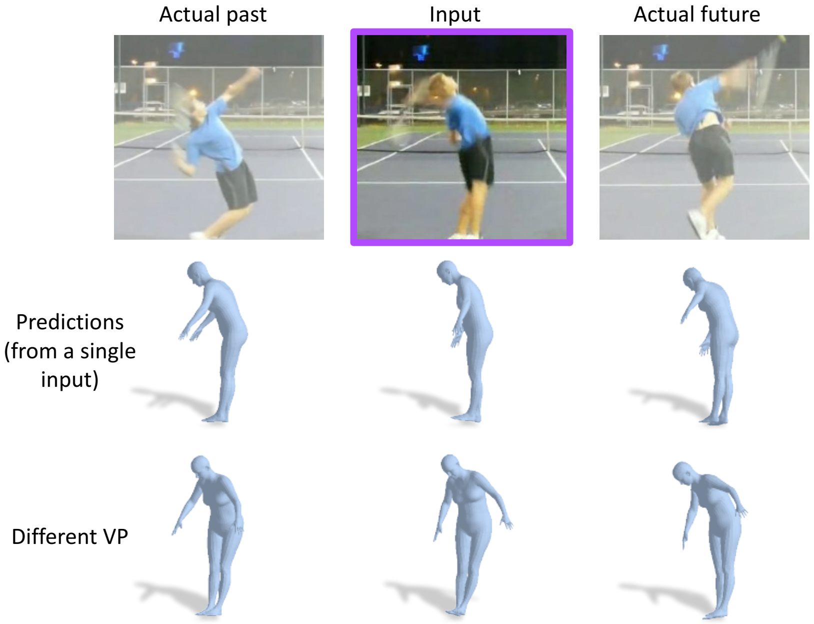


Figure 4.7: Ambiguous pose. The tennis player's pose in the input, center image is difficult to disambiguate between hunched forward verses arched backward due to the motion blur. This makes it challenging for our model to recover accurate dynamics predictions from the single image.

Chapter 5

Conclusion

In this thesis, we have presented a number of advances towards learning to predict human behavior. In particular, in Chapter 2, we presented methods for predicting future events. In Chapter 3, we presented methods for predicting how events will evolve, specifically by predicting human motion. We further personalized predictions based on the identities of the moving agents. Both for events and for motion prediction, we largely operated in an overhead view representation of sports game play. In Chapter 4, we worked towards making predictions from appearance alone, where we presented a framework for learning 3D human dynamics from video that is capable of outputting smooth 3D predictions of a persons pose and shape in videos. And further, from a single image, it can recover the current 3D pose and shape as well as its 3D motions in the immediate surrounding moments. With these methods, there is still much room for improvement. Below we discuss some of the challenges that remain, and some possible future directions.

Long-term prediction: Accurately predicting farther in the future is harder. While our predictions for player motion in Chapter 3 are somewhat long-term, extending up to 8 seconds, our predictions in Chapter 4, for example, are only within an 11-frame window around a given snapshot. And, our event-based predictions are largely for the next future event. However, in any of these cases, we could extend our model to include auto-regressive prediction for longer-term forecasting, as has been shown to be successful for generating long sequences of motion with [cite the autoconditioning paper]. Only in Chapter 4, where our hallucinator relies on past and future frames, would we need to adapt our model prior to incorporating an auto-regressive framework. However, replacing our presented temporal encoder in Chapter 4 with a causal version, does enable auto-regressive predictions of future 3D human motion.

Multi-modal prediction: Naturally, the future is uncertain. Capturing this uncertainty with multi-modal predictions is an area of active research. In Chapter 3, we explicitly attempt to model the distribution of future motion with a variational layer. However, while our predictions are quite good, we are largely unsuccessful with capturing a variety of different possible futures. Recent work from [77] introduces an alternative generative framework with demonstrated success with image synthesis [78]. Adapting this method to our prediction tasks could enable multi-modal predictions.

Multi-type prediction: In this work, we present distinct methods for predicting future events and future motion. However, in real life, events and motion are intertwined. Events provide a coarse description for what occurs, whereas motion provides the information about how an event occurs. Recent work shows promise with generating player trajectories using weakly supervised macro intents [145], which capture activity information. While other recent work shows the benefit of using motion prediction as an auxiliary task for learning to recognize actions in fly motion trajectories [34].

Representation: The overhead representation of game play that we present in Chapter 2, and subsequently rely on in Chapter 3, has the advantage of capturing well the player spacing and motion, which may be more difficult to capture directly from the pixels alone. However, the drawback is that it loses all information about pose and appearance. Methods that jointly incorporate both of these representations should be able to learn to take advantage of the benefits of each when making predictions. For example, if we provide, as an additional feature to our random forest the oracle (discretized) direction a player is facing, performance significantly improves. With four direction bins, we can surpass non-expert performance and start to approach expert-level performance here the prediction accuracy is in the low 60s.

Learning without ground truth supervision: In Chapter 4, we demonstrate noticeable performance improvement by using pseudo-ground truth 2D pose labels from more than one day's worth of footage from Internet videos. While an obvious extension would be to incorporate more and better data into learning (as pose detectors improve), another direction would be to incorporate pseudo-ground truth 3D data. For example, the OpenPose [22] authors have recently released 3D triangulation from multiple single views to estimate 3D pose in multi-view video footage. As Human3.6m, the primary source of 3D ground truth data, is limited in its actions, this may be able to provide 3D supervision for a wide variety of activities.

Bibliography

[1] Triantafyllos Afouras, Joon Son Chung, and Andrew Zisserman. "Deep Lip Reading: A Comparison of Models and an Online Application". In: *Interspeech*. 2018, pp. 3514–3518.

[2] Ankur Agarwal and Bill Triggs. "Recovering 3D human pose from monocular images". In: vol. 28. 1. IEEE, 2006, pp. 44–58.

[3] Pulkit Agrawal et al. "Learning to poke by poking: Experiential learning of intuitive physics". In: 2016.

[4] I. Akhter et al. "Bilinear Spatiotemporal Basis Models". In: 2012.

[5] Alexandre Alahi et al. "Social lstm: Human trajectory prediction in crowded spaces". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 961–971.

[6] A. Alahi et al. "Social LSTM: Human Trajectory Prediction in Crowded Spaces". In: *CVPR*. 2016.

[7] Thiemo Alldieck et al. "Optical flow-based 3d human motion estimation from monocular video". In: *GCPR*. Springer. 2017, pp. 347–360.

[8] Thiemo Alldieck et al. "Video Based Reconstruction of 3D People Model". In: *CVPR*. 2018.

[9] Mykhaylo Andriluka et al. "2D Human Pose Estimation: New Benchmark and State of the Art Analysis". In: *CVPR*. June 2014.

[10] Salman Aslam. *YouTube by the Numbers*. `https://www.omnicoreagency.com/youtube-statistics/`. Accessed: 2018-05-15. 2018.

[11] Timur Bagautdinov et al. "Social scene understanding: End-to-end multi-person action localization and collective activity recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 4315–4324.

[12] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling". In: 2018.

[13] Michael Beetz et al. "Aspogamo: Automated sports game analysis models". In: vol. 8. 1. 2009, pp. 1–21.

[14] Horesh Ben Shitrit et al. “Tracking multiple people under global appearance constraints”. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE. 2011, pp. 137–144.

[15] Alina Bialkowski et al. “Large-scale analysis of soccer matches using spatiotemporal tracking data”. In: *Data Mining (ICDM), 2014 IEEE International Conference on*. IEEE. 2014, pp. 725–730.

[16] Federica Bogo et al. “Keep it SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image”. In: *ECCV*. 2016.

[17] Paulo Vinicius Koerich Borges, Nicola Conci, and Andrea Cavallaro. “Video-based human behavior understanding: a survey”. In: vol. 23. 11. IEEE, 2013, pp. 1993–2008.

[18] Samuel R. Bowman et al. “Generating Sentences from a Continuous Space”. In: *CoRR* abs/1511.06349 (2015). arXiv: `1511.06349`. URL: `http://arxiv.org/abs/1511.06349`.

[19] Matthew Brand, Nuria Oliver, and Alex Pentland. “Coupled hidden Markov models for complex action recognition”. In: *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. IEEE. 1997, pp. 994–999.

[20] Judith Bütepage et al. “Deep representation learning for human motion prediction and classification”. In: *CVPR*. IEEE. 2017, p. 2017.

[21] A. Butt and R. Collins. “Multi-target Tracking by Lagrangian Relaxation to Min-Cost Network Flow”. In: *CVPR*. 2013.

[22] Zhe Cao et al. “OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields”. In: *arXiv preprint arXiv:1812.08008*. 2018.

[23] Zhe Cao et al. “Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields”. In: *CVPR*. 2017.

[24] Joao Carreira et al. “Human Pose Estimation with Iterative Error Feedback”. In: *CVPR*. 2016.

[25] Yu-Wei Chao et al. “Forecasting Human Dynamics from Static Images.” In: *CVPR*. 2017, pp. 3643–3651.

[26] J. Charles et al. “Personalizing Human Video Pose Estimation”. In: *CVPR*. 2016.

[27] J. Chen et al. “Learning Online Smooth Predictors for Realtime Camera Planning using Recurrent Decision Trees”. In: *CVPR*. 2016.

[28] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. “Real-time tracking of non-rigid objects using mean shift”. In: *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*. Vol. 2. IEEE. 2000, pp. 142–149.

[29] Huseyin Coskun et al. “Long Short-Term Memory Kalman Filters: Recurrent Neural Estimators for Pose Regularization.” In: *ICCV*. 2017.

[30] Rishabh Dabral et al. “Learning 3D Human Pose from Structure and Motion”. In: *ECCV*. 2018.

[31] Rishabh Dabral et al. “Structure-Aware and Temporally Coherent 3D Human Pose Estimation”. In: 2018.

[32] Z. Deng et al. “Factorized Variational Autoencoders for Modeling Audience Reactions to Movies”. In: *CVPR*. 2017.

[33] Emily L Denton et al. “Unsupervised learning of disentangled representations from video”. In: *NeurIPS*. 2017, pp. 4414–4423.

[34] Eyrun Eyjolfsdottir et al. “Learning recurrent representations for hierarchical behavior modeling”. In: *arXiv preprint arXiv:1611.00094* (2016).

[35] Panna Felsen, Pulkit Agrawal, and Jitendra Malik. “What will Happen Next? Forecasting Player Moves in Sports Videos”. In: *ICCV*. IEEE Computer Society, 2017, pp. 3362–3371.

[36] Panna Felsen, Patrick Lucey, and Sujoy Ganguly. “Where Will They Go? Predicting Fine-Grained Adversarial Multi-Agent Motion using Conditional Variational Autoencoders”. In: *ECCV*. IEEE Computer Society, 2018.

[37] Chelsea Finn, Ian Goodfellow, and Sergey Levine. “Unsupervised learning for physical interaction through video prediction”. In: *NeurIPS*. 2016, pp. 64–72.

[38] David F. Fouhey et al. “From Lifestyle VLOGs to Everyday Interactions”. In: *CVPR*. 2018.

[39] David Fouhey and C Zitnick. “Predicting object dynamics in scenes”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 2019–2026.

[40] Katerina Fragkiadaki et al. “Learning Visual Predictive Models of Physics for Playing Billiards”. In: *arXiv preprint arXiv:1511.07404* (2015).

[41] Katerina Fragkiadaki et al. “Recurrent network models for human dynamics”. In: *ICCV*. 2015, pp. 4346–4354.

[42] Ruohan Gao, Bo Xiong, and Kristen Grauman. “Im2Flow: Motion Hallucination from Static Images for Action Recognition”. In: *CVPR*. 2018.

[43] Rohit Girdhar et al. “Detect-and-Track: Efficient Pose Estimation in Videos”. In: *CVPR*. 2018.

[44] Ross Girshick. “Fast R-CNN”. In: *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*. ICCV ’15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 1440–1448. ISBN: 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.169. URL: http://dx.doi.org/10.1109/ICCV.2015.169.

[45] Kristen Grauman, Gregory Shakhnarovich, and Trevor Darrell. “Inferring 3d structure with a statistical image-based shape model”. In: *ICCV*. IEEE. 2003, p. 641.

[46] Karol Gregor et al. “DRAW: A Recurrent Neural Network For Image Generation”. In: vol. abs/1502.04623. 2015. arXiv: 1502.04623. URL: http://arxiv.org/abs/1502.04623.

[47] Peng Guan et al. “Estimating human shape and pose from a single image”. In: *ICCV*. IEEE. 2009, pp. 1381–1388.

[48] Marc Habermann et al. *ReTiCaM: Real-time Human Performance Capture from Monocular Video*. 2018.

[49] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Second. Cambridge University Press, ISBN: 0521540518, 2004.

[50] Kaiming He et al. “Identity mappings in deep residual networks”. In: *ECCV*. Springer. 2016, pp. 630–645.

[51] Mir Rayat Imtiaz Hossain and James J Little. “Exploiting temporal information for 3D human pose estimation”. In: *ECCV*. Springer. 2018, pp. 69–86.

[52] De-An Huang and Kris M Kitani. “Action-reaction: Forecasting the dynamics of human interaction”. In: *Computer Vision–ECCV 2014*. Springer, 2014, pp. 489–504.

[53] Yinghao Huang et al. “Towards accurate marker-less human shape and pose estimation over time”. In: *International Conference on 3D Vision (3DV)*. 2017, pp. 421–430.

[54] E. Insafutdinov et al. “ArtTrack: Articulated Multi-Person Tracking in the Wild”. In: *CVPR*. 2017.

[55] Catalin Ionescu et al. “Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments”. In: vol. 36. 7. 2014, pp. 1325–1339.

[56] Ashesh Jain et al. “Car that knows before you do: Anticipating maneuvers via learning temporal driving models”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 3182–3190.

[57] Ashesh Jain et al. “Structural-RNN: Deep learning on spatio-temporal graphs”. In: *CVPR*. 2016, pp. 5308–5317.

[58] A. Jain et al. “Recurrent Neural Networks for Driver Activity Anticipation via Sensory-Fusion Architecture”. In: *ICRA*. 2016.

[59] Hanbyul Joo, Tomas Simon, and Yaser Sheikh. “Total capture: A 3d deformation model for tracking faces, hands, and bodies”. In: *CVPR*. 2018, pp. 8320–8329.

[60] Angjoo Kanazawa et al. “End-to-end Recovery of Human Shape and Pose”. In: *CVPR*. 2018.

[61] Angjoo Kanazawa* et al. “Learning 3D Human Dynamics from Video”. In: *CVPR*. IEEE Computer Society, 2019.

[62] Vasiliy Karasev et al. “Intent-aware long-term prediction of pedestrian motion”. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 2543–2549.

[63] K. Kim et al. “Motion Fields to Predict Play Evolution in Dynamic Sports Scenes”. In: *CVPR*. 2010.

[64] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[65] D Kingma et al. “Semi-Supervised Learning with Deep Generative Models”. In: *NIPS*. 2014.

[66] Kris Kitani et al. “Activity forecasting”. In: *Computer Vision–ECCV* (2012), pp. 201–214.

[67] Julian Francisco Pieter Kooij et al. “Context-based pedestrian path prediction”. In: *Computer Vision–ECCV 2014*. Springer, 2014, pp. 618–633.

[68] Hema S Koppula and Ashutosh Saxena. “Anticipating human activities using object affordances for reactive robotic response”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 38.1 (2016), pp. 14–29.

[69] Y. Koren, R. Bell, and C. Volinksy. “Matrix factorization techniques for recommender systems”. In: *Computer* 42.8 (2009).

[70] Henrik Kretzschmar, Markus Kuderer, and Wolfram Burgard. “Learning to predict trajectories of cooperatively navigating agents”. In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 4015–4020.

[71] Harold W. Kuhn. “The Hungarian Method for the assignment problem”. In: *Naval Research Logistics Quarterly* 2 (1955), pp. 83–97.

[72] Tian Lan, Tsung-Chuan Chen, and Silvio Savarese. “A hierarchical representation for future action prediction”. In: *Computer Vision–ECCV 2014*. Springer, 2014, pp. 689–704.

[73] Ivan Laptev et al. “Learning realistic human actions from movies”. In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE. 2008, pp. 1–8.

[74] Christoph Lassner et al. “Unite the People: Closing the Loop Between 3D and 2D Human Representations”. In: *CVPR*. July 2017.

[75] H. Le et al. “Coordinated Multi-Agent Imitation Learning”. In: *ICML*. 2017.

[76] N. Lee et al. “DESIRE: Distance Future Prediction in Dynamic Scenes with Interacting Agents”. In: *CVPR*. 2017.

[77] Ke Li and Jitendra Malik. “Implicit maximum likelihood estimation”. In: *arXiv preprint arXiv:1809.09087* (2018).

[78] Kevin Li, Tianhao Zhang, and Jitendra Malik. “Diverse Image Synthesis from Semantic Layouts via Conditional IMLE”. In: *CoRR* abs/1811.12373 (2018).

[79] Yijun Li et al. “Flow-Grounded Spatial-Temporal Video Prediction from Still Images”. In: *ECCV*. 2018.

[80] Zimo Li et al. “Auto-conditioned recurrent networks for extended complex human motion synthesis”. In: *ICLR* (2018).

[81] Mude Lin et al. “Recurrent 3d pose sequence machines”. In: *CVPR*. IEEE. 2017, pp. 5543–5552.

[82] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440.

[83] Matthew Loper et al. "SMPL: A Skinned Multi-Person Linear Model". In: *SIGGRAPH Asia* (2015).

[84] Patrick Lucey et al. "Representing and discovering adversarial team behaviors using player roles". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 2706–2713.

[85] P. Lucey et al. "Representing and Discovering Adversarial Team Behaviors using Player Roles". In: *CVPR*. 2013.

[86] A. Maksai, X. Wang, and P. Fua. "What Players do with the Ball: A Physically Constrained Interaction Modeling". In: *CVPR*. 2016.

[87] Andrii Maksai, Xinchao Wang, and Pascal Fua. "What Players do with the Ball: A Physically Constrained Interaction Modeling". In: *arXiv preprint arXiv:1511.06181* (2015).

[88] Timo von Marcard et al. "Recovering Accurate 3D Human Pose in The Wild Using IMUs and a Moving Camera". In: *ECCV*. Sept. 2018.

[89] Julieta Martinez et al. "A simple yet effective baseline for 3d human pose estimation". In: *ICCV*. 2017.

[90] Michael Mathieu, Camille Couprie, and Yann LeCun. "Deep multi-scale video prediction beyond mean square error". In: *arXiv preprint arXiv:1511.05440* (2015).

[91] Dushyant Mehta et al. "VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera". In: *SIGGRAPH*. July 2017. URL: http://gvv.mpi-inf.mpg.de/projects/VNect/.

[92] John Miller and Moritz Hardt. "Stable Recurrent Models". In: *ICLR* (2019).

[93] Roozbeh Mottaghi et al. "Newtonian Image Understanding: Unfolding the Dynamics of Objects in Static Images". In: *arXiv preprint arXiv:1511.04048* (2015).

[94] Juan Carlos Niebles, Chih-Wei Chen, and Li Fei-Fei. "Modeling temporal structure of decomposable motion segments for activity classification". In: *Computer Vision–ECCV 2010*. Springer, 2010, pp. 392–405.

[95] Junhyuk Oh et al. "Action-Conditional Video Prediction using Deep Networks in Atari Games". In: *NIPS* (2015).

[96] Mohamed Omran et al. "Neural Body Fitting: Unifying Deep Learning and Model-Based Human Pose and Shape Estimation". In: *International Conference on 3D Vision (3DV)*. 2018.

[97] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. "Pixel Recurrent Neural Networks". In: vol. abs/1601.06759. 2016. arXiv: 1601.06759. URL: http://arxiv.org/abs/1601.06759.

[98] Deepak Pathak et al. "Context Encoders: Feature Learning by Inpainting". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[99] Georgios Pavlakos et al. "Learning to Estimate 3D Human Pose and Shape from a Single Color Image". In: *CVPR*. 2018.

[100] Dario Pavllo et al. "3D human pose estimation in video with temporal convolutions and semi-supervised training". In: *CVPR*. 2019.

[101] Xue Bin Peng et al. "SFV: Reinforcement Learning of Physical Skills from Videos". In: *SIGGRAPH Asia* 37.6 (Nov. 2018).

[102] Leonid Pishchulin et al. "DeepCut: Joint Subset Partition and Labeling for Multi Person Pose Estimation". In: *CVPR*. 2016, pp. 4929–4937.

[103] Ilija Radosavovic et al. "Data distillation: Towards omni-supervised learning". In: *CVPR* (2018).

[104] Vignesh Ramanathan et al. "Detecting events and key actors in multi-person videos". In: *arXiv preprint arXiv:1511.02917* (2015).

[105] MarcAurelio Ranzato et al. "Video (language) modeling: a baseline for generative models of natural videos". In: *arXiv preprint arXiv:1412.6604* (2014).

[106] Ali Rehan et al. "NRSfM using local rigidity". In: *WACV*. IEEE. 2014, pp. 69–74.

[107] Eike Rehder and Horst Kloeden. "Goal-Directed Pedestrian Prediction". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2015, pp. 50–58.

[108] Helge Rhodin et al. "General automatic human shape and motion capture using volumetric contour cues". In: *ECCV*. Springer. 2016, pp. 509–526.

[109] Mikel D Rodriguez, Javed Ahmed, and Mubarak Shah. "Action mach a spatio-temporal maximum average correlation height filter for action recognition". In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE. 2008, pp. 1–8.

[110] L. Sha et al. "Fine-Grained Retrieval of Sports Plays using Tree-Based Alignment of Trajectories". In: *arXiv:1710.02255*. 2017.

[111] Leonid Sigal, Alexandru Balan, and Michael J Black. "Combined discriminative and generative articulated pose and non-rigid shape estimation". In: *NeurIPS*. 2008, pp. 1337–1344.

[112] K. Simonyan and A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *International Conference on Learning Representations*. 2015.

[113] Karen Simonyan and Andrew Zisserman. "Two-stream convolutional networks for action recognition in videos". In: *Advances in Neural Information Processing Systems*. 2014, pp. 568–576.

[114] K. Sohn, H. Lee, and X. Yan. "Learning Structured Output Representation using Deep Conditional Generative Models". In: *NIPS*. 2015.

[115] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. "UCF101: A dataset of 101 human actions classes from videos in the wild". In: *arXiv preprint arXiv:1212.0402* (2012).

[116] STATS. *https://www.stats.com/sportvu-basketball/*. URL: `https://www.stats.com/sportvu-basketball/`.

[117] Shan Su et al. "Social Behavior Prediction from First Person Videos". In: *CoRR* abs/1611.09464 (2016). arXiv: `1611.09464`. URL: `http://arxiv.org/abs/1611.09464`.

[118] Jun Kai Vince Tan, Ignas Budvytis, and Roberto Cipolla. "Indirect Deep Structured Learning for 3D Human Shape and Pose Prediction". In: *BMVC*. 2017.

[119] Matthew Trumble et al. "Deep autoencoder for combined human pose estimation and body model upscaling". In: *ECCV*. 2018, pp. 784–800.

[120] Hsiao-Yu Tung et al. "Self-supervised Learning of Motion Capture". In: *NeurIPS*. 2017, pp. 5242–5252.

[121] Raquel Urtasun, David J Fleet, and Pascal Fua. "3D people tracking with Gaussian process dynamical models". In: *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. Vol. 1. IEEE. 2006, pp. 238–245.

[122] Gül Varol et al. "BodyNet: Volumetric Inference of 3D Human Body Shapes". In: *ECCV*. 2018.

[123] Ruben Villegas et al. "Neural Kinematic Networks for Unsupervised Motion Retargetting". In: *CVPR*. 2018.

[124] Carl Vondrick, Donald Patterson, and Deva Ramanan. "Efficiently Scaling up Crowdsourced Video Annotation". In: *International Journal of Computer Vision* (2012), pp. 1–21. ISSN: 0920-5691.

[125] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. "Anticipating the future by watching unlabeled video". In: *arXiv preprint arXiv:1504.08023* (2015).

[126] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. "Generating videos with scene dynamics". In: *Advances In Neural Information Processing Systems*. 2016, pp. 613–621.

[127] Jacob Walker, Abhinav Gupta, and Martial Hebert. "Dense optical flow prediction from a static image". In: *ICCV*. 2015, pp. 2443–2451.

[128] Jacob Walker, Arpan Gupta, and Martial Hebert. "Patch to the future: Unsupervised visual prediction". In: *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE. 2014, pp. 3302–3309.

[129] Jacob Walker et al. "An uncertain future: Forecasting from static images using variational autoencoders". In: *European Conference on Computer Vision*. Springer. 2016, pp. 835–851.

[130] Jacob Walker et al. "The Pose Knows: Video Forecasting by Generating Pose Futures". In: *ICCV*. 2017.

[131] Bastian Wandt, Hanno Ackermann, and Bodo Rosenhahn. “3d reconstruction of human motion from monocular image sequences”. In: *TPAMI* 38.8 (2016), pp. 1505–1516.

[132] S. Wang and C. Fowlkes. “Learning Optimal Parameters for Multi-Target Tracking”. In: *BMVC*. 2016.

[133] Xinyu Wei et al. “Forecasting events using an augmented hidden conditional random field”. In: *Computer Vision–ACCV 2014*. Springer, 2014, pp. 569–582.

[134] Xinyu Wei et al. “Predicting shot locations in tennis using spatiotemporal data”. In: *Digital Image Computing: Techniques and Applications (DICTA), 2013 International Conference on*. IEEE. 2013, pp. 1–8.

[135] Shiuh-Ku Weng, Chung-Ming Kuo, and Shu-Kang Tu. “Video object tracking using adaptive Kalman filter”. In: *Journal of Visual Communication and Image Representation* 17.6 (2006), pp. 1190–1208.

[136] Jiajun Wu et al. “Single Image 3D Interpreter Network”. In: *ECCV*. 2016.

[137] Weipeng Xu et al. “MonoPerfCap: Human Performance Capture From Monocular Video”. In: vol. 37. 2. New York, NY, USA: ACM, May 2018, 27:1–27:15. DOI: `10.1145/3181973`. URL: `http://doi.acm.org/10.1145/3181973`.

[138] Tianfan Xue et al. “Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks”. In: *NeurIPS*. 2016, pp. 91–99.

[139] K. Yamaguchi et al. “Who are you with and where are you going?” In: *CVPR*. 2011.

[140] Yi Yang and Deva Ramanan. “Articulated human detection with flexible mixtures of parts”. In: vol. 35. 12. IEEE, 2013, pp. 2878–2890.

[141] Serena Yeung et al. “Every Moment Counts: Dense Detailed Labeling of Actions in Complex Videos”. In: *arXiv preprint arXiv:1507.05738* (2015).

[142] Alper Yilmaz, Omar Javed, and Mubarak Shah. “Object tracking: A survey”. In: *Acm computing surveys (CSUR)* 38.4 (2006), p. 13.

[143] Yisong Yue et al. “Learning fine-grained spatial models for dynamic sports play prediction”. In: *2014 IEEE International Conference on Data Mining*. IEEE. 2014, pp. 670–679.

[144] Andrei Zanfir, Elisabeta Marinoiu, and Cristian Sminchisescu. “Monocular 3D pose and shape estimation of multiple people in natural scenes-the importance of multiple scene constraints”. In: *CVPR*. 2018, pp. 2148–2157.

[145] Eric Zhan et al. “Generating Multi-Agent Trajectories using Programmatic Weak Supervision”. In: *arXiv preprint arXiv:1803.07612* (2018).

[146] Weiyu Zhang, Menglong Zhu, and Konstantinos G Derpanis. “From actemes to action: A strongly-supervised representation for detailed action understanding”. In: *CVPR*. 2013, pp. 2248–2255.

[147] S. Zheng, Y. Yue, and P. Lucey. “Generating Long-term Trajectories Using Deep Hierarchical Networks”. In: *NIPS*. 2016.

[148] Shizhe Zhou et al. “Parametric reshaping of human bodies in images”. In: *SIGGRAPH*. ACM. 2010, p. 126.

[149] Tinghui Zhou et al. “View synthesis by appearance flow”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 286–301.

[150] Xiaowei Zhou et al. “Sparseness meets deepness: 3D human pose estimation from monocular video”. In: *CVPR*. 2016, pp. 4966–4975.

[151] Yipin Zhou and Tamara L Berg. “Temporal Perception and Prediction in Ego-Centric Video”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 4498–4506.