Discriminative Predictive Analysis for Goal Prediction

BY

ADITYA CHAUDHARY
B.Tech., Uttar Pradesh Technical University, 2012

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Chicago, 2017

Chicago, Illinois

Defense Committee:

Brian Ziebart, Computer Science, Chair and Advisor
Miloš Žefran, Electrical and Computer Engineering
Robert Kenyon, Computer Science

I dedicate this work to Arunima Gupta who put faith in me when I didn't. My parents who were always behind me and my sister who continuously irritated me.

ACKNOWLEDGMENTS

I sincerely thank Professor Brian Ziebart who had the patience to answer all my trivial questions and oversaw the development of the work. It wouldn't have been possible without his help and suggestions.

I thank Professor Robert Kenyon, his interest and eagerness to learn about my work really humbled me. I would like to thank Professor Milos Zefran, taking his class is the only reason I got to do this. I also would like to thank them for being a part of my thesis committee.

Finally, I convey my thanks to the team of Christopher Schultz and Sanket Gaurav in the robotics lab for collecting the data that I was able to use for my research.

AC

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

RSS	Residual Sum of Squares
AI	Artificial Intelligence
CV	Cross Validation
UIC	University of Illinois at Chicago

SUMMARY

Goal prediction has always been of interest for researchers. With the advent of robots in human life and humans working so closely with them, it is of paramount importance that goal prediction be looked at more closely to improve the human-robot interaction and prevent industrial accidents.

The work that has been done in this field has been generative in nature. This thesis looks at discriminative goal prediction where it predicts the final goal of a robotic arm given its partial trajectory. Data for the experiment was collected from human teleoperation using Baxter robot and Microsoft Kinect. The features for learning are extracted from the partial trajectory. A logit model is used to fit the training data and predict from the test data. Both accuracy and log loss are used as evaluation criteria to see how well the model performs. The results verify the effectiveness of the discriminative goal prediction.

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

Robots have become a regular part of our lives. The current AI wave has made it possible for robots to be involved in our day to day lives. Although the main operation for robots is still in industrial operations where human involvement is not safe (Nagatani et al., 2013) or in settings where human strength is not enough to complete the operation(Harada et al., 2005), a comparatively new area for robotics has arisen in medical surgery(Lendvay et al., 2013) where precise movement is required and human movement doesn't compare very well with the robot's precision. As robots have started working alongside humans in industrial settings, target prediction for robotic movement is becoming an important and necessary feature.

In order to avoid accidents in industrial settings target prediction for robotic movements is extremely important for the human-robot interaction to go smoothly. It's important for robot to know where the human will be so it can provide assistance and likewise it's important for human to know where the robot will finally go. Most of the work done in the area of target prediction has been generative which is computationally expensive.

1.2 Thesis Statement

This thesis aims at Discriminative goal prediction of a robotic hand from partial trajectory. A discriminative model directly learns the labels from input and hence is better computationally than generative models. Firstly, data is transformed as per the requirements for the model. For this problem partial trajectory is taken till 60 time steps. Different features are extracted from the partial trajectory and then the model is trained using those features. Finally the model is tested on the test data to check how well it performs. Data for this study was gathered from teleoperation performed on a Baxter robot using Microsoft Kinect as a sensor to capture human motion pose.

1.3 Thesis Organization

This document is as outlined as follows: The first chapter gives a basic introduction to the problem and proposed solution. The second chapter is the background and it addresses previous work in the area of target prediction and is both regression and classification based. The third chapter looks at what machine learning models and evaluation measures have been used in the work.

The fourth chapter talks about different methodologies or features extracted from partial trajectory used in the model. the fifth chapter showcases the results and discussion and finally the sixth chapter is the conclusion.

CHAPTER 2

BACKGROUND

2.1 Robotic Teleoperation for Data Collection

Teleoperation is controlling of a machine from a distance. It started with mechanical linkages and cables but wireless communication i.e. Bluetooth, Wifi and advanced sensors have made it possible for us to use teleoperation from a safe perspective. Also, since usage of teleoperation is in niche areas where human life can be at risk. These advancement have come like a boon.

There are two types of teleoperation:

• Unilateral: Where human master controller gives an instruction and robot slave controller follows

• Bilateral: Where there is a force/haptic feedback from the robot slave controller to master controller

Point cloud data of a human teleoperator is obtained using Microsoft Kinect. To overlay a digital skeleton model on the human teleoperator's captured depth camera data OpenNI framework is applied to Kinect data. The OpenNI Skeleton model has 105 data pointsin total where 15 skeleton points each have w,x,y,z rotational and x,y,z translation data.

Data collection for this project was done using teleoperation on a Baxter robot using Microsoft Kinect as a sensor to generate human motion pose which the robot followed.

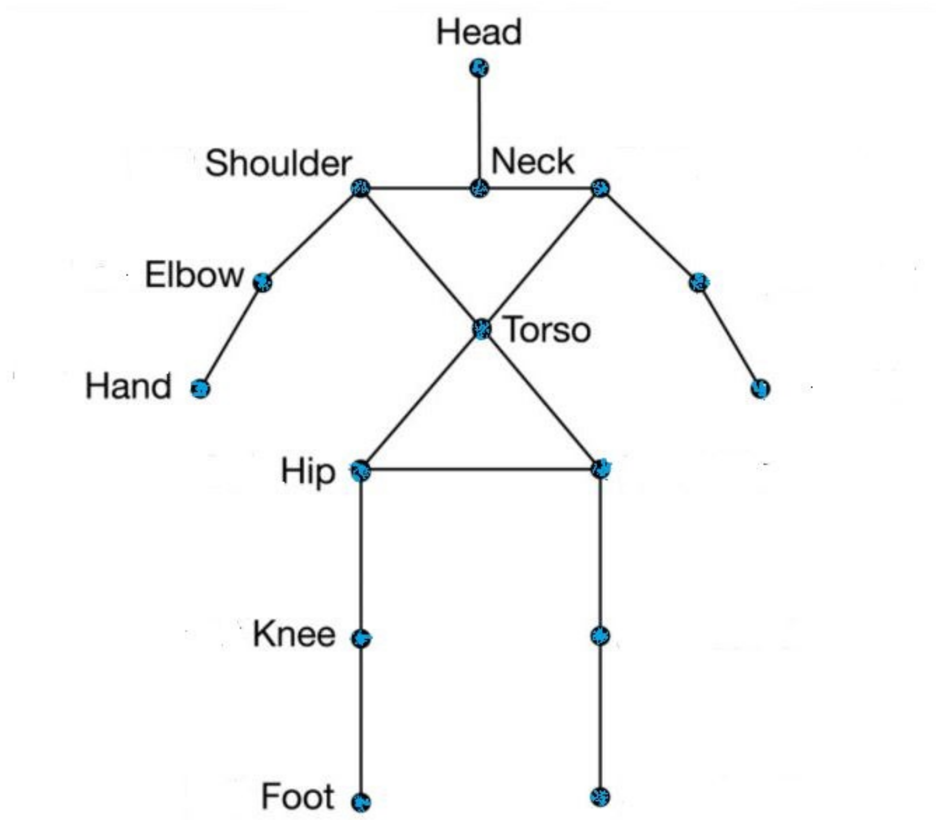


Figure 1: OpenNI skeleton model where each point has 3 translational and 4 rotational data

2.2 Generative and Discriminative Models

Generative models use the joint probability p(x,y) to learn a model(Ng and Jordan, 2001). The prediction is done with Bayes Theorem where $p(y \mid x)$ is calculated and the most likely label is picked.

$$p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)} \tag{2.1}$$

A Discriminative model learns a direct map for class labels from input x or models $p(y \mid x)$ directly.

A very good example is, given a training set machine learning algorithms try to find a decision boundary for any number of examples present. With the test set they try to classify on what side that example falls. Cats and dogs for instance. This method is discriminative. In generative, the model tries to build a model depending on different features of examples. Different face structure of cats and dogs. Post this model building test set is compared against these models and classified accordingly. Generative models are computationally expensive as compared to discriminative models.

2.3 Previous Work

Some work has been done in the field of target prediction using different methods; normative kinematic law(Lank et al., 2007), simple regression(Asano et al., 2005), Kalman filters(Aydemir et al., 2013), neural networks(Biswas et al., 2013), inverse control theory(Ziebart et al., 2012) and Kinematic template matching(Pasqual and Wobbrock, 2014). Some of them were target agnostic while others were target aware. More mathematical approaches though are easily adaptable but their complex nature makes it difficult to implement.

2.3.1 Regression based work

A relationship between peak velocity of a trajectory and its final distance was found in a prior research(Takagi et al., 2002). Using that Asano et al(Asano et al., 2005) used Linear Regression model to predict final distance using peak velocity.

$$D = a + PV.b \tag{2.2}$$

Keuning Van Oirschot and Houtsma(Oirschot and Houtsma, 2001) found out that doubling the distance of a trajectorys peak velocity roughly predicts the total length. These models being too trivial werent able to give high quality predictions.

A different approach taken by Lank et al(Lank et al., 2007) involves predictive equation using normative Kinematic laws. This work is based on minimum jerk law(Flash and Hogan, 1985) and has two steps. Firstly, a quadratic curve is fit to the velocity-distance profile using least-square regression. An extrapolation of this curve gave the endpoint which was the non-zero x-intercept. The numerical instability of the whole process usually overshot the goal and hence coefficients were applied from a pre-calculated table. The work was further improved in a later work by removing the coefficients to correct endpoint after extrapolation. The following criterion was used to check the stability:

$$\frac{l_{\mathrm{n}} - l_{\mathrm{n\text{-}1}}}{l_{\mathrm{n}}} < 0.2 \tag{2.3}$$

l_n is the length of current prediction

l_{n-1} is the length of previous prediction

Using this new algorithm, Ruiz et al(Ruiz and Lank, 2009)(Ruiz and Lank, 2010) provided a single shot version which gives a single value after the ratio of completed movement to predicted movement crosses a threshold. With this algorithm 51.4% accuracy is achieved after 90% of the motion is completed while it reduces to 33.7% when 85% of the motion is completed.

2.3.2 Classification Based Work

This work comprised of target aware techniques, which is prediction of targets from a set of possible targets.

Murata et al(Murata, 1998) devised a cumulative score for prediction. At each interval, the angle between the movement direction vector and vector connecting to each target from the current vector. This angle is added to the cumulative score for each target and the target with lowest cumulative score is taken to be the predicted target.

The approach used by Ziebart et al(Ziebart et al., 2012) attached probabilities to targets using inverse optimal control and Bayes rule. An instantaneous pointing state was represented by position, velocity, acceleration and jerk combined at a given time. The changes in velocity are represented as transitions between such states. Using inverse optimal control, target locations of previously collected movements is used to create a probabilistic model of pointing movements. Bayes rule is then applied over assumed uniform prior distribution to provide a probability for each target. They could attain an accuracy of 60% when 90% of the movement has been completed.

Schultz et al(Schultz et al., 2017) developed a goal predictive teleoperation model using Inverse Optimal Control to predict the intended goal of a trajectory and then autonomously completes the predicted task.

The most recent work in classification used neural network(Biswas et al., 2013) and Kalman filter(Aydemir et al., 2013). In the first technique authors trained the network on velocity, acceleration and angle of movement to predict the corrective sub-movement phase. Using that prediction, the direction of movement is calculated and nearest target is chosen as the intended target. In second technique, based on angle and distance to each target authors used Kalman filters. Using this model probabilities are assigned to each target and one with largest probability is selected. An accuracy of approximately 60% was achieved on evaluating both able-bodied and motor-impaired users.

Most of these models are generative in nature and learn the joint probability first to predict the labels. Also, they use typically one feature. The work done in this thesis aims at discriminative model and learns the labels directly from input and combines multiple kinematic features for predictions.

As Vapnik(Vapnik, 1998) puts *one should solve the [classification] problem directly and never solve a more general problem as an intermediate step [such as modeling] P($x|y$).*

CHAPTER 3

MACHINE LEARNING MODELS USED AND EVALUATION MEASURES

3.1 Linear Regression

Linear Regression(James et al., 2013) is a machine-learning algorithm, which takes use of the linear relationship between predictor variable and quantitative response. The linear relationship is approximate.

$$Y = a_0 + a_1 X \tag{3.1}$$

a_0 and a_1 are two unknown constants that represent the intercept and slope of the linear model. Most common approach to predict a_0 and a_1 is minimizing the least squares. Y is predictor variable and X is quantitative response.

Let $\hat{y}_i = a_0 + a_1\hat{x}_i$ be the prediction for Y based on i^{th} value of X. Then $e_i = y_i - \hat{y}_i$, represents the i^{th} residual, which is the difference between i^{th} observed value and i^{th} predicted value by the linear model.

The Residual Sum of Squares (RSS) is given by

$$RSS = {e_1}^2 + {e_2}^2 + {e_3}^2......{e_n}^2 \tag{3.2}$$

Or equivalently as:

$$RSS = (y_1 - a_0 - a_1x_1)^2 + (y_2 - a_0 - a_1x_2)^2 + (y_3 - a_0 - a_1x_3)^2 + ...(y_n - a_0 - a_1x_n)^2 \tag{3.3}$$

We can solve for a_0 and a_1

$$a_1 = \frac{\sum_{i=1}^{n}(x_{\text{i}}-\bar{x})(y_{\text{i}}-\bar{y})}{\sum_{i=1}^{n}(x_{\text{i}}-\bar{x})^2} \tag{3.4}$$

$$a_0 = \bar{y} - \hat{a}_1\bar{x} \tag{3.5}$$

where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_{\text{i}}$ and $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_{\text{i}}$ are the sample means. These two equations define the least squares coefficient estimates for linear regression.

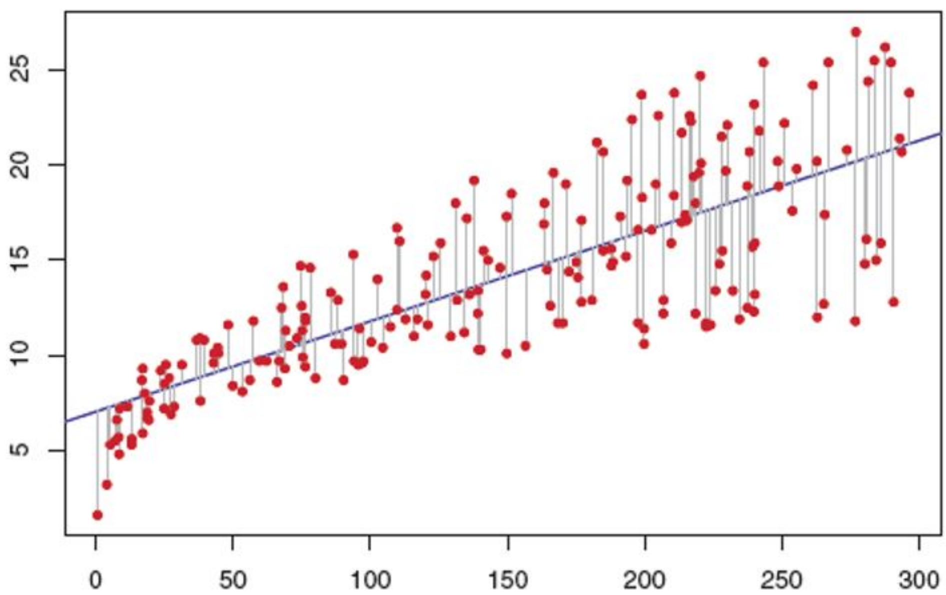


Figure 2: Least Squares fit for regression

3.2 Logistic Regression

While linear regression works good (with some reservations) with quantitative data. Predictions on qualitative data cant be done with linear regression. For example, if someone asks a question, do you like ice-cream. The response can either be yes and no or 0 and 1, it cant be 0.7. This comes under classification problem which is tackled easily by logistic regression.

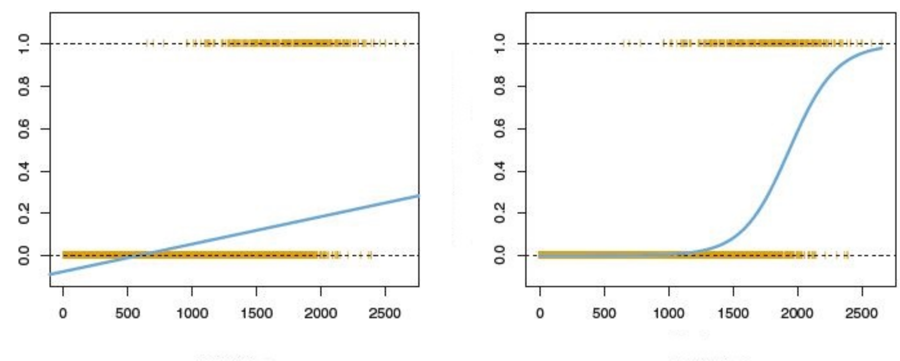


Figure 3: Figure showing estimation using Linear Regression on left and Logistic regression on right

For Logistic Regression(James et al., 2013) we use logistic function which predicts between 0 and 1 for all values of X.

$$p(X) = \frac{e^{\mathrm{a}_0+\mathrm{a}_1\mathrm{X}}}{(1+e^{\mathrm{a}_0+\mathrm{a}_1\mathrm{X}})} \tag{3.6}$$

After some manipulation we get:

$$\frac{p(X)}{1-p(X)} = e^{\mathrm{a_0+a_1X}} \tag{3.7}$$

To get logit model we take logarithm on both sides

$$log(\frac{p(X)}{1-p(X)}) = a_0 + a_1X \tag{3.8}$$

We notice that logistic regression model has a logit which is linear in X. Also, we can have multiple predictors in logit model.

$$p(X) = \frac{e^{\mathrm{a_0+a_1X_1+a_2X_2+a_3X_3+.....a_nX_n}}}{(1+e^{\mathrm{a_0+a_1X_1+a_2X_2+a_3X_3+.....a_nX_n}})} \tag{3.9}$$

The coefficients $\mathrm{a_0}$ and $\mathrm{a_1}$ are unknown and are predicted using training data. For prediction we use maximum likelihood function for its better statistical properties. The basic idea for using maximum likelihood for logistic regression is predicting values of $\mathrm{a_0}$ and $\mathrm{a_1}$ so that predicted probability $\hat{p}(x_\mathrm{i})$ comes as close to the observed probability. Other coefficients can be calculated in the same manner.

$$l(a_0, a_1) = \prod_{i:y_\mathrm{i}=1} p(x_\mathrm{i}) \prod_{i:y_{\mathrm{i}'}=1} 1 - p(x_{\mathrm{i}'}) \tag{3.10}$$

3.3 Cross Validation

Cross Validation(James et al., 2013) is a model evaluation technique.In the absence of a test set, various techniques can be used for estimation using available training data. In cross validation a part of training data is held out to be used as test data.

3.3.1 Validation Set

In Validation set, training data is split in two: training and testing. The split can be done randomly but usually a 70:30 or 80:20 ratio is maintained for training and testing data. The model is trained on training dataset and tested on testing dataset to check how well the model performs.

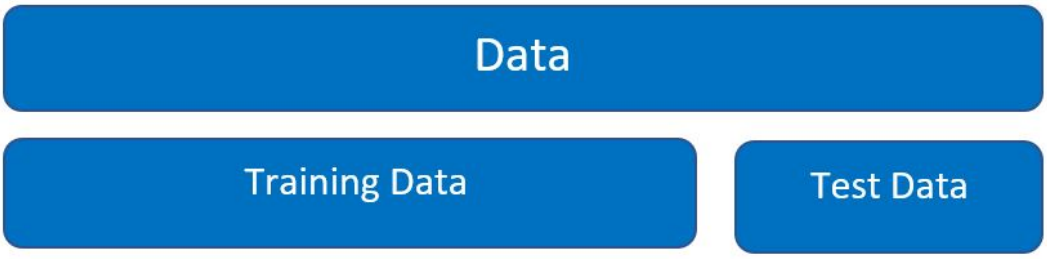


Figure 4: Validation Set split of data

There is a problem with validation set. The estimation of test set is highly dependent on which observation is included in the test set and which is included in the training set. Also, since the whole training set is not included in the training of model as the data is split. The trained model will perform worse than if the whole data was used.

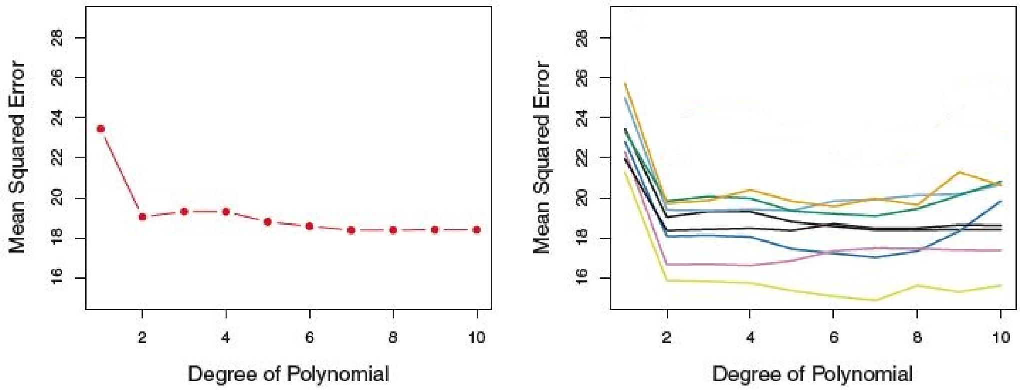


Figure 5: Different validation sets producing different errors on same dataset

3.3.2 Leave-One-Out Cross-Validation

Leave-One-Out Cross-Validation or LOOCV is very closely related to validation set. It was introduced to overcome the drawbacks of Validation Set. LOOCV also involves splitting of data into two parts but instead of splitting it in a 70:30 or 80:20 ratio. Only one observation is taken as test set and rest n-1 observations are taken as training set. MSE for the test set can be calculated by:

$$MSE_1 = (y_1 - \hat{y_1})^2 \tag{3.11}$$

This MSE is unbiased but a poor approach as it is highly variable as only one observation is used for test set. Hence, this procedure is repeated for every single observation in the data set. Consequently MSE_2, MSE_3...MSE_n are calculated using the same approach. The final estimate is the average of these MSE's.

$$CV = \frac{1}{n}\sum_{i=1}^{n} MSE_i \tag{3.12}$$

This technique is better than validation set in the sense that the number of observations used are greater than used in validation set. The learning of the model is better in this case. Also, as compared to Validation set there is no randomness in the training/test split. So results come out to be same.

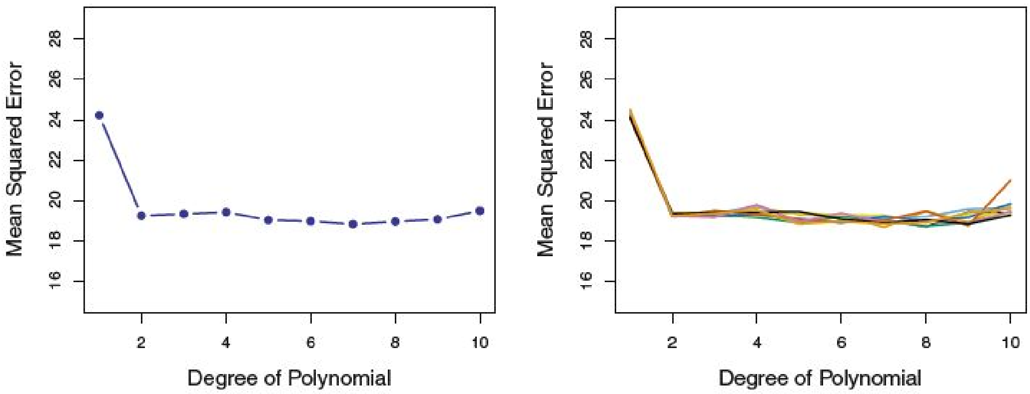


Figure 6: Same results produced when using LOOCV

3.3.3 K-fold Cross Validation

K-fold Cross Validation involves randomly dividing the data set into k groups or folds of equal size. Training is done on k-1 sets and testing is done on remaining 1 fold. This computation is done k times and each time a different set is chosen for testing. K-fold CV is calculated using:

$$CV = \frac{1}{k}\sum_{i=1}^{k} MSE_i \tag{3.13}$$

This technique is better than LOOCV as it is computationally less expensive than LOOCV. If data set is large then LOOCV has to be run n times and that can be very expensive computationally. K-fold is a better option in that case.

3.4 Accuracy and Log loss

Confusion matrix is the most common criteria used in classification tasks. Various measures are:

True Positive(TP) are the positive labels which are positively classified

False Positive(FP) are the negative labels which are positively classified

False Negative(FN) are the positive labels which are negatively classified

True Negative(TN) are the negative labels which are negatively classified

Accuracy is measure of correct predictions by total predictions.

$\frac{TP+TN}{total}$

Log loss is a performance metric for evaluating predictions of probabilities for different classes. Wrong predictions are punished according to their confidence of prediction. A classifier is penalized more if it predicted a class 0 with probability 0.86 but is penalized less if it did the same with probability 0.32 while the right class was 1.

Log loss for multi-class logistic regression can be calculated using:

$$logloss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{i,j} log(p_{i,j}) \tag{3.14}$$

$y_{i,j}$ is the binary variable with expected labels

$p_{i,j}$ is the classifier's probability output

N is number of instances

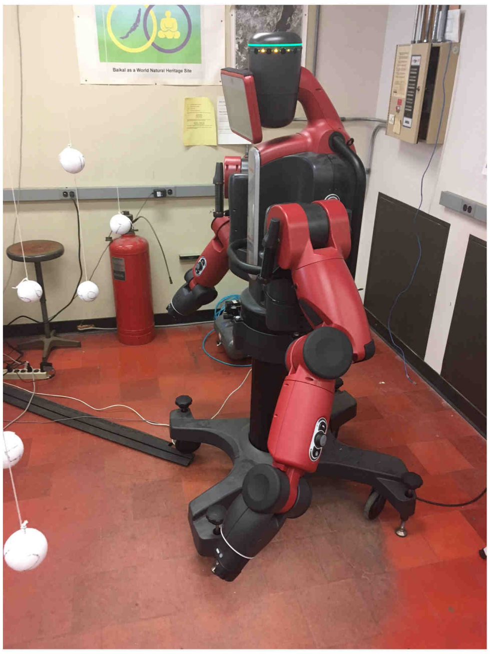
M is the number of different labels

CHAPTER 4

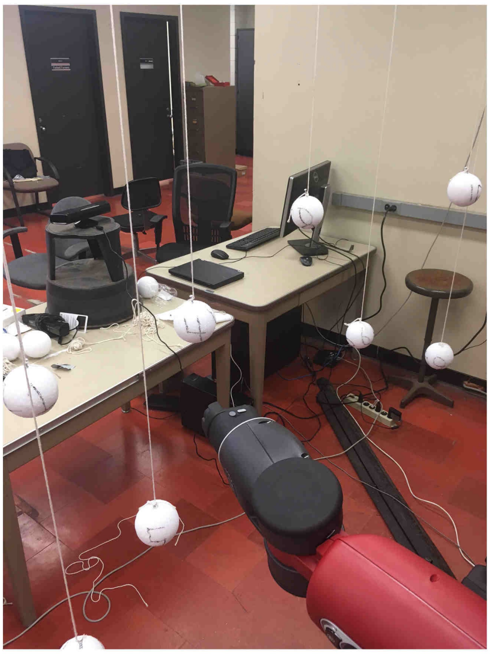
METHODOLOGIES USED IN MODEL

4.1 Data Collection and Data Modeling

Teleoperation was used for Data Collection. A total of ten different end-effector goal locations are present in the environment, five for each of the Baxter arms. All the goals are approximately four-inch diameter spheres numbered to reflect the goal index they are associated with and are visually shown in the testing space around the robot. A human demonstrator's OpenNI skeleton data is collected by placing a Kinect camera in front of the robot testing space. The Kinect camera is positioned in such as way so that demonstrator has a visual of both Baxter robot and the goals. The human demonstrator is given five seconds to prepare after being told what the objective goal is prior to the start of any goal test sequence through a graphics display. The sequence starts after this countdown. Totally, 18 sets of testing data from human demonstrators were completed and each demonstrator contributed 60 data points.

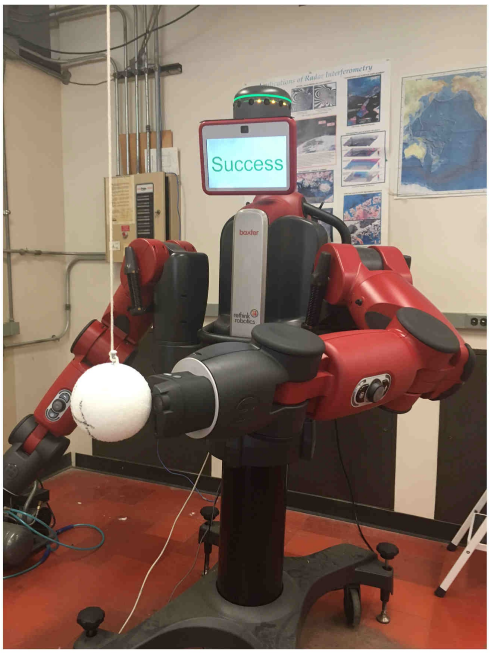


a. Starting Neutral Position



b. Teleoperate Arm Towards Goal

c. At Goal



d. Results Displayed

Figure 7: Steps involved in Data Collection

The collected data had time steps, 3 spatial coordinates and 4 rotational coordinates as well as some control data to be used for the previous study. For the purpose of this study only time steps and spatial coordinates were needed. Rest of the data was dropped. The big concern now was modeling the data so it could be used for learning as well as testing. There were 1184 csv's in total where 1 csv had the whole trajectory data. Flatten function of pandas was used on the all the columns to make one big column and then data was transposed to make a row. All the csv's were decomposed to one csv where each row consisted of the whole trajectory data.

The figure below shows the trajectories and their starting point as well as positions of different goals in 3-D space.

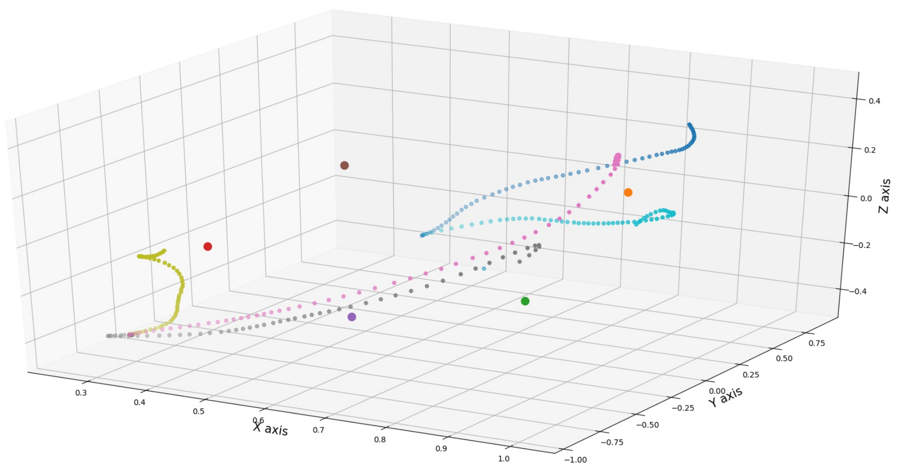


Figure 8: Figure of partial trajectories with goal positions

4.2 Cosine of Angle

Cosine similarity was used to calculate the cosine of angle between the vector which was at the last point of the trajectory and each of the 10 goals using the relation:

$$\cos\theta = \frac{\vec{a}.\vec{b}}{|\vec{a}||\vec{b}|} \tag{4.1}$$

Using the maximum values or minimum values of cosine brings a human bias into the model. Selecting all the cosine values as a feature improves the model's predictive power.

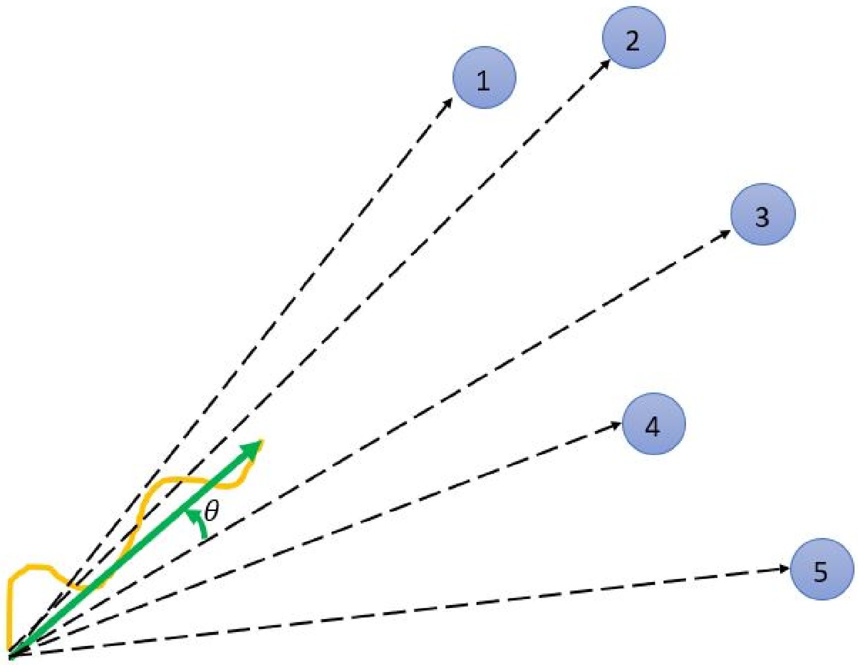


Figure 9: Angle between the trajectory and goals

4.3 Distance using Peak Velocity

In this research Asano et al conducted an experiment with human participants to find a relationship between peak velocity and distance. They analyzed the aforementioned relationship by undertaking tasks which involved pointing to virtual objects using a controller with six degrees of freedom tracker. The position data was collected by tracker. They found out that peak velocity has a linear relationship with distance and devised an algorithm following these rules:

- Any user roughly always follows a straight line to target
- The total distance covered has a linear relationship with the peak velocity of the trajectory

$$D = a + PV.b \tag{4.2}$$

where D is total distance

PV is Peak Velocity

a and b are constants

The assumption here being that this trajectory is one single continuous movement from start point to end-point.

This algorithm was incorporated in building the logit model. The instantaneous velocity was calculated from the spatial data where we had x, y, z coordinates for every time step. Euclidean distance was calculated using:

$$d = \sqrt[2]{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{4.3}$$

Instantaneous distance between different time steps divided by total time gave us instantaneous velocity. The maximum out of these up to the set partial trajectory point which was set as 60 time steps gave us the peak velocity.

To calculate a and b this data was fit onto a linear regression model. The intercept of the model gave us a and coefficient gave us b. Every trajectory had a different intercept and coefficient and hence total distance was predicted for every trajectory. Post finding the distance, this was added to the first point(P_x, P_y, P_z) of trajectory in the direction of the last vector of partial trajectory.

The last vector (L_x, L_y, L_z) had to be normalized first:

$$L = \sqrt[2]{(L_x)^2 + (L_y)^2 + (L_z)^2} \tag{4.4}$$

Now,

$L_{xi} = \frac{L_x}{L}$

$L_{yi} = \frac{L_y}{L}$

$L_{zi} = \frac{L_z}{L}$

The new position becomes:

$N_x = P_x + (L_{xi} * D)$

$N_y = P_y + (L_{yi} * D)$

$N_z = P_z + (L_{zi} * D)$

The minimum distance from this new position (N_x, N_y, N_z) was calculated to all the possible goals and the Euclidean distance returned was chosen as a feature. This feature when trained with the model was not adding much to the accuracy of the model. Hence an error term to correct the total distance was added. The new algorithm became:

$$D = a + PV.b + e \tag{4.5}$$

This error term was the residual or the minimum distance that was calculated in the previous step. Total distance was again calculated using this algorithm and new positions were again calculated. Finding the distance to each goal and then using that as a feature improved the accuracy for the model.

4.4 Curve Fit and Extrapolation of Data

Lank et al used extrapolation for predicting gesture length. The process involves fitting a polynomial curve to the data points. In (Oirschot and Houtsma, 2001) (Flash and Hogan, 1985) it has been empirically defined that end-effector usually always follow a straight line. Using this information, extrapolating from the fitted curve becomes easy. It has been noted as well that extrapolation is an unstable process numerically, hence, the lowest degree polynomial is used for fitting.

Curve fitting is done using a spline function which is made up of sum of B-spline and uses least squares method. B-spline function passes through control points called knots and is a combination of flexible bands which passes through them to create smooth curves. A number of points in these functions are used for management and creation of complex shapes and surfaces . A B-spline of order n is a piecewise polynomial which is defined over n+1 locations called knots.

B-splines are useful in the fact that any spline function of order n can be expressed as a linear combination of B-splines on a given set of knots.

All the velocity-time curve for the trajectory were done using Univariate spline function of scikit-learn. This curve was extrapolated and the point where it crossed the x-axis was the total time taken. Multiplying this time with average velocity gave the total distance covered.

$$D = V.T \tag{4.6}$$

V is average velocity

T is total time

This total distance is added to the first point of trajectory in the direction of last point of trajectory using the method discussed in previous section. The distance is again calculated between this new point N and all the possible goals and the distance is chosen. Using this distance as a feature didn't help with the prediction. Hence a residual term was added to the calculated total distance.

$$D = V.T + e \tag{4.7}$$

e is the residual term which is the minimum distance between the new point N and all the possible goals

A new point M was calculated again in the direction of the last point of trajectory and it's distance to the goals was chosen as a feature.

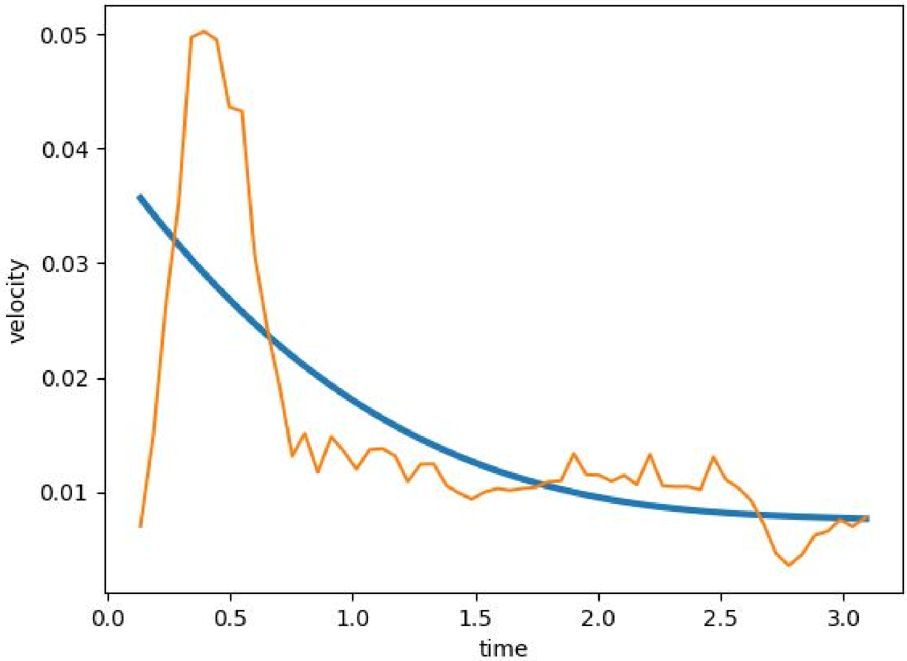


Figure 10: Example of Curve fit for one of the trajectories

A problem with the curve fit was initial lag in the robotic hand. The velocity-time curve was coming out very different than it should have been.

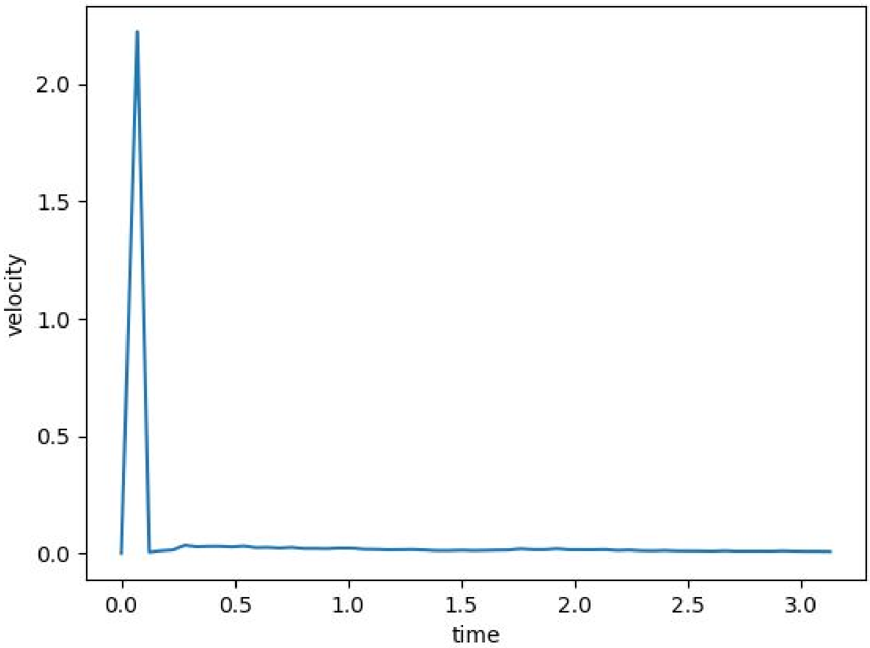


Figure 11: Problem in velocity-time curve due to initial lag

To take care of this problem, first two velocities were dropped from the data frame which gave a better smoother curve.

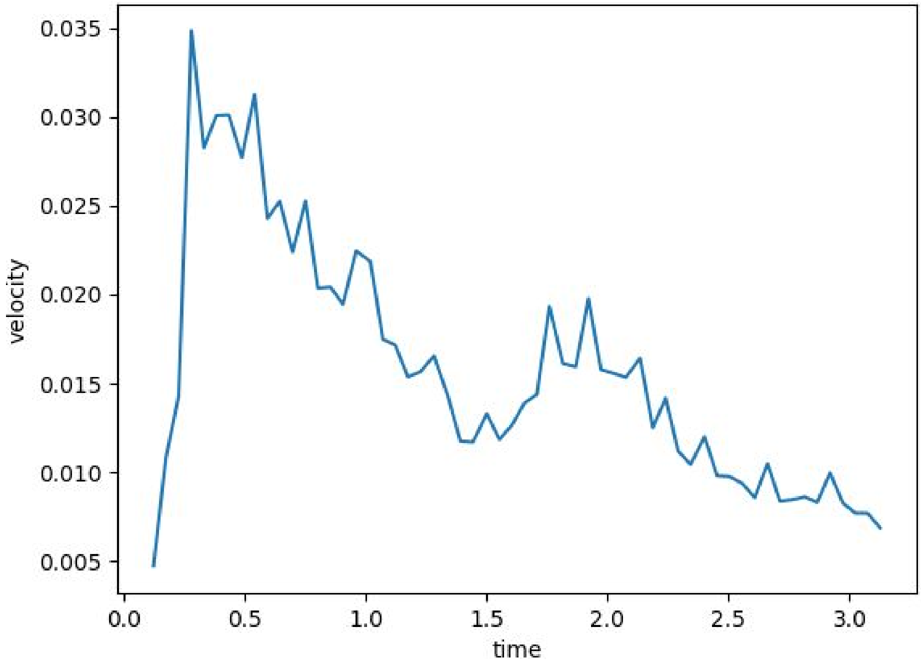


Figure 12: Corrected figure after dropping the values

CHAPTER 5

RESULTS

Firstly all the features were fit into the model individually and learning was done using K-fold cross validation.

Feature	Accuracy	Log Loss	Confidence Interval
Cosine of angle	44.22%	1.45	44.22 ± 0.026
Distance using peak velocity	55.35%	1.36	55.35 ± 0.027
Curve fit	52.52%	1.40	53.51 ± 0.029
All features	57.89%	1.06	57.89 ± 0.024

When all of these features were used together. An accuracy of **57.89%** was achieved. Target agnostic methods like KTM(Pasqual and Wobbrock, 2014) and KEP(Ruiz and Lank, 2009) show 21.5% and 9.2% accuracy of hitting the target after 90% of movement is done in 2-D, while they achieve 18.8% and 13.3% accuracy in 1-D. In (Lank et al., 2007) after 80% of movement 42.4% of target predictions are within $\pm$0.5W of target center.

The log loss for the model came to be **1.06**. For a sense of this number, the uniform distribution over 10 goals is $\log_2(10)$ which is 3.32. That is the log loss value achieved if all the goals have uniform probabilities distributed over them.

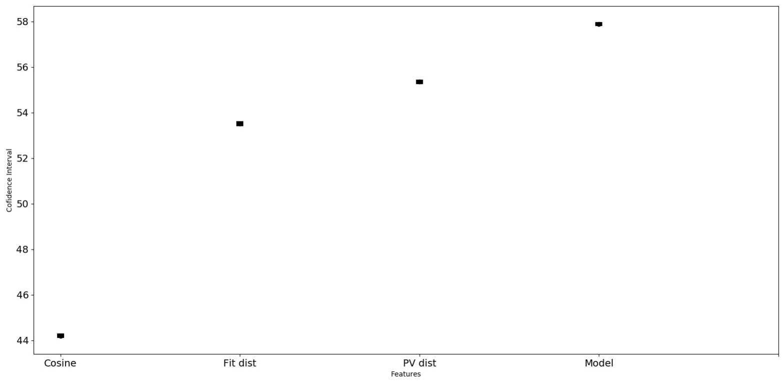


Figure 13: Confidence Interval for features and model

CHAPTER 6

CONCLUSION AND FUTURE WORK

This work shows that it is possible to use partial trajectory to predict the final goal with good accuracy and log loss which is much better than just chance. This discriminative goal prediction performed better than many other generative work done in the field of goal prediction.

For future work, firstly some other features can be extracted from the partial trajectory to improve the model. Training of this model on more varied data will help it in improving its predictive power. Secondly it can be used on teleoperation, sensor noise poses to be a big problem for teleoperation and hence using this model, the final goal can be predicted from partial trajectory for the robot to finish its task.

CITED LITERATURE

[Asano et al. , 2005] Asano, T., Sharlin, E.and Kitamura, Y., Takashima, K., and Kishino, F.: Predictive interaction using the delphian desktop. In Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '05), 2005.

[Aydemir et al. , 2013] Aydemir, G., Langdon, P., and Godsill, S.: User target intention recognition from cursor position using kalman filter. In Proceedings of the 7th Int'l Conference on Universal Access in Human-Computer Interaction (UAHCI '13). Las Vegas, Nevada. 2013.

[Biswas et al. , 2013] Biswas, P., Aydemir, G., Langdon, P., and Godsill, S.: Intent recognition using neural networks and kalman filters. In Proceedings of Human-Computer Interaction and Knowledge Discovery (HCI-KDD '13). Maribor, Slovenia (July 1-3, 2013), 2013.

[Flash and Hogan, 1985] Flash, T. and Hogan, N.: The coordination of arm movements: an experimentally confirmed mathematical model. In The Journal of neuroscience : the official journal of the Society for Neuroscience, 1985.

[Harada et al. , 2005] Harada, K., Kajita, S., Saito, H., Morisawa, M., Kanehiro, F., Fujiwara, K., Kaneko, K., , and Hirukawa, H.: A humanoid robot carrying a heavy object. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pages 1712–1717, 2005.

[James et al. , 2013] James, G., Witten, D., Hastie, T., , and Tibshirani, R.: An introduction to statistical learning. In Springer vol 6, 2013.

[Lank et al. , 2007] Lank, E., C., Y.-C.N., and Ruiz, J.: Endpoint prediction using motion kinematics. In Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI '07). San Jose, California, 2007.

[Lendvay et al. , 2013] Lendvay, T. S., Hannaford, B., and Satava, R. M.: Future of robotic surgery. The Cancer Journal, 2013.

[Murata, 1998] Murata, A.: Improvement of pointing time by predicting targets in pointing with a pc mouse. In International Journal of Human-Computer Interaction, 1998.

[Nagatani et al. , 2013] Nagatani, K., Kiribayashi, S. O. Y., Otake, K., Yoshida, K., Tadokoro, S., Nishimura, T., Yoshida, T., Koyanagi, E., and Fukushima, M.: Emergency response to the nuclear accident at the fukushima daiichi nuclear power plants using mobile rescue robots. In Journal of Field Robotics, 30(1):4463, 2013.

[Ng and Jordan, 2001] Ng, A. and Jordan, M.: On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In Proceeding CHI '14 Proceedings of NIPS, 2001.

[Oirschot and Houtsma, 2001] Oirschot, H. and Houtsma, A.: Cursor displacement and velocity profiles for targets in various locations. In Proceedings of EuroHaptics 2001. Birmingham, United Kingdom (July 1-4, 2001), 2001.

[Pasqual and Wobbrock, 2014] Pasqual, P. T. and Wobbrock, J. O.: Mouse pointing endpoint prediction using kinematic template matching. In Proceeding CHI '14 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems Pages 743-752, 2014.

[Ruiz and Lank, 2009] Ruiz, J. and Lank, E.: Effects of target size and distance on kinematic endpoint prediction. In Technical Report CS-2009-25. Cheriton School of Computer Science. Waterloo, Ontario: University of Waterloo, 2009.

[Ruiz and Lank, 2010] Ruiz, J. and Lank, E.: Speeding pointing in tiled widgets: Understanding the effects of target expansion and misprediction. In Proceedings of the ACM Conference on Intelligent User Interfaces (IUI '10). Hong Kong, China, 2010.

[Schultz et al. , 2017] Schultz, C., Gaurav, S., Monfort, M., Zhang, L., and Ziebart, B. D.: Goal-predictive robotic teleoperation from noisy sensors. In Submitted to IEEE ICRA 2017 IEEE, 2017.

[Takagi et al. , 2002] Takagi, R., Kitamura, Y., Naito, S., and Kishino, F.: A fundamental study on error-corrective feedback movement in a positioning task. In Proceedings of the 5th Asia-Pacific Conference on ComputerHuman Interaction (APCHI '02). Beijing, China, 2002.

[Vapnik, 1998] Vapnik, V.: Statistical learning theory. 1998.

[Ziebart et al. , 2012] Ziebart, B., Dey, A., and Bagnell, J.: Probabilistic pointing target prediction via inverse optimal control. In Proceedings of the ACM Conference on Intelligent User Interfaces (IUI'12). Lisbon, Portugal, 2012.

VITA

Aditya Chaudhary

Education	M.S. Electrical and Computer Engineering University of Illinois at Chicago	2015 – 2017
	B.Tech. Electronics and Communication Engineering Uttar Pradesh Technical University	2008 – 2012
Publications	**Aditya Chaudhary** and Arunima Gupta, “Piezoelectric Based Energy Source Model for Street Lights”, IJERT, 2013.	
	Aditya Chaudhary and Akash Tyagi, “Traffic Based Dynamic Request Control System for Web Page Analysis”, IJCA, 2013.	
Experience	Software Engineering Analyst at Accenture	