Adversarial Learning for Intractable Problems

by

Wei Xing
M.S., Zhejiang University, 2013

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Chicago, 2022

Chicago, Illinois

Defense Committee:

Brian Ziebart, Chair and Advisor
Xinhua Zhang
Sathya Ravi
Anastasios Sidiropoulos
Thorsten Joachims, Cornell University

ACKNOWLEDGMENT

I should sincerely give my thanks and respect to my advisor Dr. Brian Ziebart. He is a great mentor who brings us the wonderful new field of adversarial learning and inspires me with a lot of interesting directions and ideas. He is always enthusiastic, diligent and instructive. He gave me the energy when I was down and helped me all the way until the end of my Ph.D. candidate journey. My committee members Dr. Xinhua Zhang, Dr. Sathya Ravi, Dr. Anastasios Sidiropoulos and Dr. Thorsten Joachims also gave me great advises and feedback to finish my thesis.

Thanks to my labmates and collaborators, especially Kaizer Asif, Hong Wang, Sima Behpour, Rizal Fathony, Anqi Liu, Xiangli Chan, Sanket Gaurav, Vena Li, Ashkan Rezaei, Mohammad Ali Bashiri, Mathew Monfort and Zhan Shi who shared me with their great knowledge and creativity so that I could accomplish my academic works.

And without the companion of my friends Guixiang Ma, Bokai Cao, Chun-Ta Lu, Bowen Dong, Honghan Shuai and Chenwei Zhang, I can never lived a such a colorful life in Chicago.

Finally, thanks to my mother and father who supported me all my paths to the doctoral degree.

WX

CONTRIBUTIONS OF AUTHORS

My advisor Dr. Brian Ziebart participated in all my works. He helped me from proposing the problem, modeling, debugging codes to figuring out what results need to be presented.

Chapter 3 introduces the basic knowledge and ideas of adversarial learning. I participated in (1) which was one of the early works in this field. Kaiser Asif was the primary author who wrote the major codes and the paper. Dr. Sima Behpour and me joined the discussion of the main model and we helped in background study and doing the comparison experiments. I also participated in (2) which was the first paper that brought the important technique double oracle in solving adversarial learning problems with complex structure. Dr. Hong Wang was the primary author who built the major code structure. I participated in expanding the framework to the problem of discounted cumulative gain, including writing the corresponding codes, paper and doing the experiments.

Chapter 4 contains two major works. The first one is an extension of paper (3). In (3), Dr. Sima Behpour was the primary author who wrote the code, run the experiments and wrote a large part of the paper. I contributed in figuring out the best responses in the double oracle methods and wrote the major theoretical parts of sections ”Adversarial Robust Cuts” and ”Adversarial Robust Cuts”. Chapter 4 differs from (3) as it expands the problem to multi-label cases and solves a quit different double oracle best response problem. The second one is (4) which is a extension of (5), and I was the primary author.

Chapter 5 contains another parts of (4).

Chapter 6 presented the experiements results of the models in chapter 4 and chapter 5, so it also contains contents from (3) and (4).

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

3D 3 dimensional

ADMM alternating direction method of multipliers

ADV-MC adversarial multiway cut

CRF conditional random field

k-NN k-nearest neighbor

LPR linear programming relaxation

MRF Markov random field

SVM support vector machine

SVM-ECOC support vector machine with error-correcting output codes

SVM-Struct structured support vector machine

SUMMARY

After intensive studies on basic problems such as classification and clustering, the machine learning community has reached a stage where many novel problems contain interrelated variables and the learning process usually contains complex inference sub-problems. Graphical models such as Markov random field (MRF) and maximal marginal methods such as structured support vector machine (SVM-Struct) are popular choices for solving those problems. Graphical models usually suffer great burden in computing the normalization term and marginal methods do not have Fisher consistency with many easy to use surrogate losses such as hinge loss. Adversarial learning has been proposed to overcome those shortcomings with additional benefits such as being robust towards noises and easy adaptation to different problems. It has been proved to be advanced in both theoretically and practically on many problems that have exact solutions for the sub inference problem such as multiclass classification and multilabel prediction.

On the other hand, there are many even harder problems in which the inference part can be NP-hard. This means approximation has to be applied to produce an efficient solution. With the additional errors introduced to the computation process, the theoretical guarantees works on normal cases may no longer hold. In this work, we will explore the performance of adversarial learning on solving these kinds of hard problems.

We will focus on two problems. The first one is multiclass classification with pairwise features. This problem represents a group of hard problems that are independently identically distributed (i. i. d) assumption of the data is broken. The inference problem is equivalent to a multiway cut problem and

it is NP-hard. We applied adversarial learning on it and reformulated the sub-problem into an integer linear programming problem. The second one is the weighted 3-dimensional matching problem which is also NP-hard. This problem is one of the problems whose output has complex structures. In this problem, we use a different way to solve it by reducing the problem to the marginal space and moving the NP hardness to the prediction stage. We compared the results with state-of-art methods and got better results from adversarial learning for both of the problems. More work will be done to explore whether and how adversarial learning can make better use of inexact results to get to a good optimal point.

CHAPTER 1

INTRODUCTION

With the increasing common use of machine learning, people encountered more and more problems that no longer only contains linear data but also complex structured data where variables are interrelated. These problems play important roles in applications such as computer vision (6), natural language processing (7), computational biology, and other areas (8; 9). Among those learning tasks, some of them have certain restricted relationships and structures (e.g., chains, trees, and other low-treewidth structures) do have efficient inference algorithms. The most general of these—binary-valued associative Markov networks (10) and the special case of attractive pairwise relationships (11)—use minimum graph cuts (12) for inference and maximum margin methods (13; 14) for training.

Unfortunately, many problems on structured data are intractable and people have to use approximation methods to solve them. When we use these approximation methods inside some well formed learning methods based on exact solutions, many guarantees and good properties of those methods may no longer hold. Actually, learning can fail in providing acceptable results even with an approximate inference method with rigorous approximation guarantees. (15). There are several aspects can cause this result. Firstly, approximate methods may reduce the expressivity of an underlying model by making it impossible to choose parameters that reliably give good predictions. Secondly, approximation can respond to parameter changes in such a way that standard learning algorithms are misled.

Besides the lose of theoretical guarantees when using approximation inference, many existed method have some drawbacks even if they work on exact inference. For example, exponentiated potential fields

models such as conditional random field (CRF) (16; 17) have high time complexity and cannot be used to solve large data sets. Maximum margin methods such as SVM-Struct that use hinge surrogate loss (18; 19) cannot reach Fisher consistency (20; 21).

Here we try to use adversarial learning on some intractable problems. Adversarial methods are based on game theory and try to learn a predictor or generator against an adversary. One of the hot topics using this idea are deep generative models (22). There are also other works tried to predict well under intentional disturbance or pollution of the training data (23). The adversarial learning this work focus on a series of methods that assume the inaccuracy of constructing true distributions from limited training data, and instead learn against an adversary that only follow a few statistics got from training data. (24; 25).

Unlike SVM-Struct and some other methods that use hinge loss surrogate that can be quite loose, or even providing meaningless performance guarantees in practice. Adversarial learning seeks to tighten this gap between training objective and evaluative loss function for structured prediction tasks. It takes the form of a zero-sum game between a predictor trying to minimize an additive loss over predicted variables and an adversarial training label approximator that seeks to maximize this same loss. It provides loss bounds that are always meaningful since the game outcome is always within the range of the evaluation loss function. On the other hand, adversarial learning have Fisher consistency (26) on a wide range of loss functions, which guarantees the model converges to the right distribution as more and more training samples are involved. All this great properties leads to the success of adversarial models on a lot of problems that provide exact solutions, such as cost-sensitive classification (1), active learning (27), classification problems with zero-one loss (28), ordinal regression (29) and chain structures (30).

Here we discover the properties of adversarial learning through solving two problems. The first one is multiclass classification with pairwise features. When pairwise features are introduced to every pair of the samples, the problem becomes much harder than the case when samples are independent and is intractable (31). In this problem, the training samples themselves are highly correlated. The second one is 3-dimensional matching, where the the inference problem of finding maximal 3D matching is MAX SNP-complete which implies it is NP-hard (32). In this problem, the predicted labels are highly correlated.

This work extends the other two works that use this minimax perspective to do structure predition on binary classification with pairwise features (3) and bipartite matching (5). Adversarial learning also achieved great performance on classification problems with zero-one loss (28), ordinal regression (29), more general cost-sensitive losses (1), multivariate losses (2), and chain structures (30).

CHAPTER 2

BACKGROUND

(This chapter contains contents in paper "Arc: Adversarial robust cuts for semi-supervised and multi-label classification. (3)", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 1905-1907). It also contains contents in paper "Adversarial Learning for 3D Matching. (4)", In Conference on Uncertainty in Artificial Intelligence (pp. 869-878). PMLR. The copyright policy for author reusing of them can be found in Appendix A)

2.1 Notation and Learning Task

We consider n predicted variables, $\mathbf{y} = (y_1, \ldots, y_n)$, chosen from a fixed set of labels $y_i \in \mathcal{Y}, \forall i \in [n]$, where $[n] = \{1, \ldots, n\}$. We denote the corresponding random variables for these label variables using capitalization, $\mathbf{Y} = (Y_1, \ldots, Y_n)$, and denote vectors and multivariate variables in bold. We denote given information or side information variables using a single vector, $\mathbf{x} \in \boldsymbol{\mathcal{X}}$, with a corresponding random variable denoted as $\mathbf{X}$. (Strict sub-portions of $\mathbf{x}$ may be relevant to each variable y_i, but for notational simplicity, we do not denote such partitions in our formulation.)

Our task in this setting is to make predictions for $\mathbf{y}$ given an input $\mathbf{x}$ and a set of m training example pairs, $(\mathbf{x}^{(j)}, \mathbf{y}^{(j)})_{j \in [m]}$, where we index training examples using a parenthetical superscript notation whenever necessary to disambiguate between different examples or denote this distribution as $\tilde{P}(\mathbf{X}, \mathbf{Y})$. Aiding in this task are a set of features relating the input variables to the predicted variables and to one another. We generically denote these feature vectors as $\boldsymbol{\phi}_{\mathbf{c}}(\mathbf{y}_{\mathbf{c}}, \mathbf{x})$ for relationships over variables in

some subset of the $\mathbf{y}$ variables denoted by $\mathbf{c} \in \mathcal{C} \subseteq 2^{[n]}$. For a subset $\mathbf{c} = \{c_1, \ldots, c_l\}$ which contains l variables, $\mathbf{y_c} = \{y_{c_1}, \ldots, y_{c_l}\}$ is the corresponding set of label values for the variables in the subset. For unary features only related to each sample, they are simply $\boldsymbol{\phi}_i(y_i, \mathbf{x})$. For pairwise relationships between y_i and y_j that also incorporate input variables, this reduces to feature functions denoted as $\boldsymbol{\phi}_{i,j}(y_i, y_j, \mathbf{x})$.

For many datasets, variables that are closely related to one another tend to have the same label. For example, pixels with similar characteristics in the same region of an image tend to belong to the same image segment. To capture this property, we define pairwise features that reflect the similarity when two variables have different labels, and use a generalized Potts model (33) to penalize assignments that do not have the same label across the edges , which is equivalent to only care the features that reflect the difference for pairs that have the same label :

$$\boldsymbol{\phi}_{i,j}(y_i, y_j, \mathbf{x}) = \mathbf{1}_{y_i = y_j}(y_i, y_j)\boldsymbol{\sigma}_{i,j}(y_i, y_j, \mathbf{x}), \tag{2.1}$$

where $\mathbf{1}()$ is a indicator function whose value is 1 only if the corresponding logical expression is true. And $\sigma_{i,j}$ is the atom feature.

2.2 Markov Networks and Intractability

Estimating the conditional probability of label variables using a Markov network is one powerful approach to the structured prediction task. Markov networks can be written as log-linear models when their densities are positive. A Markov network has the following probability distribution:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} e^{\Psi(\mathbf{y},\mathbf{x})}, \tag{2.2}$$

where the potential function Ψ decomposes into a set of potential functions over subsets of the $\mathbf{y}$ variables, $\Psi(\mathbf{y},\mathbf{x}) = \sum_{\mathbf{c}\in\mathcal{C}} \psi_{\mathbf{c}}(\mathbf{y}_{\mathbf{c}},\mathbf{x})$, with these subset potentials defined as $\psi_{\mathbf{c}}(\mathbf{y}_{\mathbf{c}},\mathbf{x}) = \theta_{\mathbf{c}} \cdot \phi_{\mathbf{c}}(\mathbf{y}_{\mathbf{c}},\mathbf{x})$ using a vector of estimated weights $\theta_{\mathbf{c}}$ that is specific to each $\mathbf{c}$. Parameter sharing with clique $\mathbf{c}'$, $\phi_{\mathbf{c}} = \phi_{\mathbf{c}'}$, can be employed to reduce the effective number of learned parameters of the model. The structure of these potentials corresponds to an undirected graphical model in which the variables in set $\mathbf{c}$ are connected by undirected edges, forming cliques in the graph.

Another direction for realizing tractable Markov networks exploits potential functions for which maximization can be solved efficiently, even though normalization is intractable due to the large tree-widths of their graphs. Binary-valued Markov networks with non-negative pairwise potentials are one example of this. Their maximum value assignments can be obtained using minimum-cut/maximum-flow algorithms. There are also some classes of potential functions with certain "convexity" properties (34) that can be solved efficiently as min-cut problems. But in general, a pairwise feature associate markov network will have a maximum a posteriori problem that is equivalent to a multi-way cut problem.

Unfortunately, even when restricted to pairwise and unary potential functions, the most probable assignment of values, $\mathbf{y}^* = \text{argmax}_{\mathbf{y}\in\mathcal{Y}} P(\mathbf{y}|\mathbf{x})$, and the normalization term, $Z(\mathbf{x}) = \sum_{\mathbf{y}\in\mathcal{Y}} e^{\sum_{\mathbf{c}\in\mathbf{C}} \psi_{\mathbf{c}}(\mathbf{y}_{\mathbf{c}},\mathbf{x})}$, are both intractable to compute for Markov networks in general (31). Restrictions are often placed on the

potential functions so that the corresponding undirected graph has low tree-width (e.g., chains, trees), which enables efficient maximization and normalization computations (31).

2.3 Maximum Margin Learning and Loose Bounds

Maximum margin methods for learning, like the structured support vector machine (13; 18), operate by introducing a hinge loss surrogate to optimize in place of the loss function of interest, which is generally non-convex and possibly not even continuous. For structured prediction, this takes the following form:

$$\min_{\theta, \boldsymbol{\epsilon} \geq \mathbf{0}} ||\theta|| + \lambda \sum_{i} \epsilon_i \text{ such that:}$$

$$\epsilon_i \geq \max_{\mathbf{y}'} \text{loss}(\mathbf{y}', \mathbf{y}^{(i)}) + \Psi(\mathbf{y}', \mathbf{x}) - \Psi(\mathbf{y}^{(i)}, \mathbf{x}), \tag{2.3}$$

where loss() can be any loss function of interest, θ is a compact representation of all estimated potential function parameters, λ is a fixed regularization parameter, and ϵ_i is the amount of hinge loss incurred by the i^{th} training example. Conceptually, this optimization seeks to make the potential of the true label vector $\Psi(\mathbf{y}^{(i)}, \mathbf{x})$ greater than all alternatives $\Psi(\mathbf{y}', \mathbf{x})$ by a margin that depends on the (Hamming) loss between $\mathbf{y}'$ and $\mathbf{y}^{(i)}$. Since the objective of this optimization is a convex function of the model parameters, θ, standard gradient-based methods can be employed to optimize θ. Achieving low ϵ_i ensures small amounts of incurred loss on example i, since $\epsilon_i \geq \text{loss}(\hat{\mathbf{y}} = \text{argmax}_{\mathbf{y}} \Psi(\mathbf{y}), \mathbf{y}^{(i)})$.

Finding parameters θ that make the total hinge loss, $\sum_i \epsilon_i$, small can be difficult. When the size of the label space $|\mathcal{Y}|$ is much larger than the number of parameters being learned, there may not be any

non-trivial choice of $\theta \neq \mathbf{0}$ that makes the potential of $\mathbf{y}^{(i)}$ optimal. When multiple training examples are considered, reducing the hinge loss for one example tends to increase the hinge losses for other examples. Due to these issues, the hinge loss for a particular example $\mathbf{y}^{(i)}$ can be much greater than not only the actual loss of the predicted labels, $\text{loss}(\hat{\mathbf{y}}, \mathbf{y}^{(i)})$, but random guessing and the worst possible loss, $\max_{\mathbf{y}} \text{loss}(\mathbf{y}, \mathbf{y}^{(i)})$, as well. When this is the case, the hinge loss bounds provide no meaningful guarantees on the predictor's performance.

2.4 Fisher Consistency

Fisher consistency is a desirable property for a learning method requiring it to provide optimal predictions when trained from many different distributions of training data under ideal learning settings.

Definition 1. *(26) A surrogate loss function, $loss'(\cdot,\cdot)$ is **Fisher consistent** for a target loss function $loss(\cdot,\cdot)$, and set of distributions $\mathcal{P}$ if the predictor f from the space of all measurable functions satisfies:*

$$\forall P \in \mathcal{P}, f^* = \underset{f}{\operatorname{argmin}}\, \mathbb{E}_P\left[loss(f(\boldsymbol{x}), y)\right] \Rightarrow$$
$$\forall \boldsymbol{x} \in \boldsymbol{x}, f^*(\boldsymbol{x}) \in \underset{\boldsymbol{y}}{\operatorname{argmin}}\, \mathbb{E}_{P(y|\boldsymbol{x})}\left[loss(y', y)\right]. \tag{2.4}$$

Theorem 1 ((21))**.** *The hinge loss is Fisher consistent for the zero-one loss under all distributions such that the most probable label for any input value has at least half the probability, i.e, $\forall \boldsymbol{x}, \operatorname{argmax}_y P(y|\boldsymbol{x}) \geq 0.5$.*

For binary-valued prediction tasks, this set of distributions covers all possible binary distributions and Fisher consistency is maintained (35). However, for multiclass prediction tasks ($|\mathcal{Y}| > 2$), a number of distributions lead to suboptimal predictions under many kinds of extended hinge loss (21).

For a predicted label y and the target label y_i (could be structured output), one of the common used hinged loss for SVM-Struct is the loss (19)

$$\text{loss}(y, y_i) = \max(0, \max_{y \in \mathcal{Y} \setminus y_i} (\Delta(y, y_i) + < \mathbf{w}, \phi(y, x) > - < \mathbf{w}, \phi(y_i, x) >)) \tag{2.5}$$

where $\mathbf{w}$ is the SSVM parameter and $\Delta(y, y_i)$ is a measurement that $\Delta(y, y) = 0$ and $\Delta(y, y') > 0$, $\forall y \neq y'$

One counter example showing this hinge loss is not Fisher consistent can be (5):

For the Hamming loss $\text{loss}(\pi, \pi') = \frac{1}{n} \sum_{i=1}^{n} 1_{\pi_i \neq \pi'_i}(\pi_i, \pi'_i)$ used for bipartile matching learning and the potential function $\psi(y) =< \mathbf{w}, \phi(y, x) >$ is additive as $\psi(\pi) = \sum_{i=1}^{n} \psi_i(\pi_i) = \sum_{i=1}^{n} < \mathbf{w}_i, \phi_i(\pi_i, x) >$. If the empirical distribution of the permutations are $P(\pi = [1\ 2\ 3]) = 0.4; P(\pi = [2\ 3\ 1]) = 0.3$; and $P(\pi = [3\ 1\ 2]) = 0.3$. When potential function is $\psi_i(\pi_i) = 1$ if $\pi_i = i$ and 0 otherwise, we can get a Bayesian optimal permutation prediction for this distribution and an expected hinge loss of $3.6 = 0.4(3-3) + 0.3(6-0) + 0.3(6-0)$. However, the expected hinge loss is optimally minimized with a value of 3 when $\psi_i(j) = 0, \forall i, j$, which is indifferent between all permutations and is not Bayes optimal. Thus, Fisher consistency is not guaranteed.

CHAPTER 3

ADVERSARIAL LEARNING

(This chapter contains contents in paper "Adversarial Cost-Sensitive Classification. (1)", In UAI 2015 Jul 12 (pp. 92-101). It also contains contents in paper "Adversarial prediction games for multivariate losses. (2)", In Advances in Neural Information Processing Systems, 28. The copyright policy for author reusing of them can be found in Appendix A)

The motivation of minimax robust learning is to approximate the training data labels with a worst-case distribution that must still resemble training data properties (24; 25; 1). The advantage of this approach versus employing the hinge loss or other convex surrogate is that: (1) the training error on the actual loss function of interest (e.g., the Hamming loss) is upper bounded by the game value; and (2) optimizing this upper bound more closely aligns the predictor's construction with its predictive performance.

3.1 Minimax Game Formulation

We introduce the distribution $\check{\mathrm{P}}(\check{\mathbf{Y}}|\mathbf{X})$ controlled by the adversary to produce worst-case approximations of the actual training example label, $\mathbf{y}^{(\mathrm{i})}$. The adversary seeks to maximize the loss subject to certain constraints based on the training data sample, while the predictor player chooses $\hat{\mathrm{P}}(\hat{\mathbf{Y}}|\mathbf{X})$ to minimize the expected loss:

$$\min_{\hat{\mathsf{P}}(\hat{\mathbf{Y}}|\mathbf{X})\in\Delta}\ \max_{\check{\mathsf{P}}(\check{\mathbf{Y}}|\mathbf{X})\in\Delta\cap\Xi} \mathbb{E}_{\mathbf{X}\sim\tilde{\mathsf{P}};\hat{\mathbf{Y}}|\mathbf{X}\sim\hat{\mathsf{P}};\check{\mathbf{Y}}|\mathbf{X}\sim\check{\mathsf{P}}}\left[\text{loss}(\hat{\mathbf{Y}},\check{\mathbf{Y}})\right] \tag{3.1}$$

Here Δ is the simplex to make the distributions between 0 and 1 and sum up to 1. Ξ is the empirical constrains we want the adversarial to follow. And $\tilde{\mathsf{P}}$ is the empirical distribution of $\mathbf{X}$ and $\mathbf{Y}$ in the training data set.

One important difference between adversarial learning and other methods is that we no longer directly learn from training samples but against an adversary. Conceptually, the adversary seeks to construct a label distribution that is as uncertain as possible, since this forces the predictor to incur large amounts of expected loss. However, the constraints placed on the adversary to match statistics ϕ relating the actual training data labels to inputs $\mathbf{x}$ and one another (such as mean or higher order moments) prevent the adversary from doing this, and, when chosen carefully, can restrict the adversary to be highly predictable.

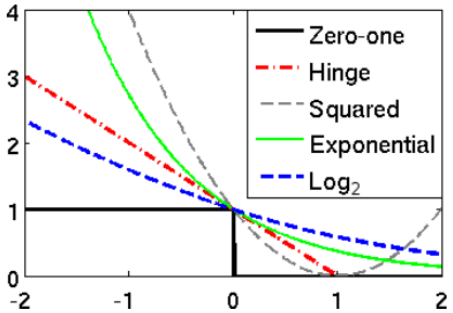

Figure 1: Different surrogate losses

In exchange of not optimization over the exact training samples, our method is able to train directly based on the exact loss (0-1 loss for classification) and still form the problem convex, and no longer needs to use surrogate losses such as those shown in Figure 1 (1).

3.2 Adversarial Cost Sensitive Classification

To better understand adversarial learning, let's take a look of the example of cost sensitive classification (1).

For cost sensitive classification, the lose for predicting a certain class is different when the correct class is different. Let's consider the problem where the cost is a fixed number for a certain prediction and real class pair, in which the lost can represented as a matrix $\mathbf{C}$ such as:

$$\mathbf{C} = \begin{bmatrix} 0 & 1 & 2 & 0 \\ 3 & 0 & 1 & 3 \\ 4 & 2 & 0 & 1 \\ 1 & 1 & 2 & 0 \end{bmatrix} \tag{3.2}$$

where $\mathbf{C}_{ij}$ is the lost when we predict class i when the real class is j. There are many researches that try to solve the problem using empirical risk minimization, including re weighting (36; 37; 38; 39), build special cost-sensitive surrogate loss (40; 41; 42; 43; 44; 45). These methods either make the problem non-convex leading to computational difficulty or target a object that can potentially got suboptimal results.

Let's define vector $\hat{\mathbf{p}}_x \triangleq \hat{P}(\hat{\mathbf{y}}|\mathbf{x}) \in \Delta$, which is the vector of conditional proprobility of all the possible $\hat{y}$ given x. Using this, we can convert (Equation 3.1) into a matrix form:

$$\begin{aligned} &\min_{\{\hat{\mathbf{p}}_x\}\in\Delta} \max_{\{\check{\mathbf{p}}_x\}\in\Delta\cap\Xi} \mathbb{E}_{\mathbf{X}\sim\tilde{P}} \left[\hat{\mathbf{p}}_{\mathbf{X}}^{\mathrm{T}}\, \mathbf{C}\, \check{\mathbf{p}}_{\mathbf{X}}\right] \\ &\text{where: } \Xi : \mathbb{E}_{\tilde{P}(\mathbf{X})\check{P}(\mathbf{Y}|\mathbf{X})} \boldsymbol{\phi}(\mathbf{Y},\mathbf{X}) = \mathbb{E}_{\tilde{P}(\mathbf{X})\tilde{P}(\mathbf{Y}|\mathbf{X})} \boldsymbol{\phi}(\mathbf{Y},\mathbf{X}) \end{aligned} \tag{3.3}$$

To solve the problem, we can put the constraint Ξ into the object function using Lagrangian multiplier θ, and since the Lagrangian is convex to θ, $\hat{\mathbf{p}}_x$ and $\check{\mathbf{p}}_x$. We have following object function (1):

$$\begin{aligned} &\min_{\{\hat{\mathbf{p}}_x\}\in\Delta} \max_{\{\check{\mathbf{p}}_x\}\in\Delta\cap\Xi} \mathbb{E}_{\mathbf{X}\sim\tilde{P}} \left[\hat{\mathbf{p}}_{\mathbf{X}}^{\mathrm{T}}\, \mathbf{C}\, \check{\mathbf{p}}_{\mathbf{X}}\right] \\ =& \max_{\{\check{\mathbf{p}}_x\}\in\Delta\cap\Xi} \mathbb{E}_{\mathbf{X}\sim\tilde{P}} \left[\min_{\{\hat{\mathbf{p}}_x\}\in\Delta} \hat{\mathbf{p}}_{\mathbf{X}}^{\mathrm{T}}\, \mathbf{C}\, \check{\mathbf{p}}_{\mathbf{X}}\right] \\ =& \min_{\theta} \mathbb{E}_{\mathbf{X}\sim\tilde{P}} \left[\max_{\{\check{\mathbf{p}}_x\}\in\Delta} \min_{\{\hat{\mathbf{p}}_x\}\in\Delta} \hat{\mathbf{p}}_{\mathbf{X}}^{\mathrm{T}}\, \mathbf{C}_{\mathbf{X},\theta}\, \check{\mathbf{p}}_{\mathbf{X}}\right] \end{aligned} \tag{3.4}$$

where $(\mathbf{C}_{\mathbf{X},\theta})_{\hat{y},\check{y}} = \mathbf{C}_{\hat{y},\check{y}} + \theta^{\mathrm{T}}(\phi(\check{y},x) - \phi(\tilde{y},x))$. If we let $\delta(\check{y},x) = \psi(\check{y},x) - \psi(\tilde{y},x) = \theta^{\mathrm{T}}(\phi(\check{y},x) - \phi(\tilde{y},x))$ the matrix $\mathbf{C}_{\mathbf{X},\theta}$ will look like:

$$\mathbf{C}_{\mathbf{X},\theta} = \begin{bmatrix} 0+\delta_1 & 1+\delta_2 & 2+\delta_3 & 0+\delta_4 \\ 3+\delta_1 & 0+\delta_2 & 1+\delta_3 & 3+\delta_4 \\ 4+\delta_1 & 2+\delta_2 & 0+\delta_3 & 1+\delta_4 \\ 1+\delta_1 & 1+\delta_2 & 2+\delta_3 & 0+\delta_4 \end{bmatrix} \tag{3.5}$$

We can solve (Equation 3.4) by two steps. Firstly, we can solve the innper problem

$$\max_{\{\check{\mathbf{p}}_x\}\in\Delta}\min_{\{\hat{\mathbf{p}}_x\}\in\Delta}\hat{\mathbf{p}}_{\mathbf{X}}^{\mathrm{T}}\,\mathbf{C}_{\mathbf{X},\theta}\,\check{\mathbf{p}}_{\mathbf{X}} \tag{3.6}$$

given particular θ. Secondly, when inner problem (Equation 3.6) is solved, we can use gradient descent or any other method to find the best θ.

The inner problem of (Equation 3.6) is a minimax game which can be solved using linear programming (46). Since $\psi(\tilde{y}, x)$ is just constant added to every element of matrix $\mathbf{C}_{\mathbf{X},\theta}$, and without it, the equilibrium of the game will be the same, so actually, for solving the inner problem, we can directly solve the game matrix:

$$\mathbf{C} = \begin{bmatrix} 0+\psi_1 & 1+\psi_2 & 2+\psi_3 & 0+\psi_4 \\ 3+\psi_1 & 0+\psi_2 & 1+\psi_3 & 3+\psi_4 \\ 4+\psi_1 & 2+\psi_2 & 0+\psi_3 & 1+\psi_4 \\ 1+\psi_1 & 1+\psi_2 & 2+\psi_3 & 0+\psi_4 \end{bmatrix} \tag{3.7}$$

In the following content, we may just refer the game matrix to (Equation 3.7). And for the outside parameter θ, we can use any subgradient methods to get them (1).

when we do prediction, we put the new x and trained θ in and solve the game again. Form the $\hat{\mathbf{p}}_x$, we find the one with highest probability as the prediction result.

Comparing with existing methods, which typically minimize a convex approximation of the cost-sensitive loss evaluated on available training data, the adversarial cost sensitive classificatio directly minimizes the actual cost-sensitive loss evaluated on an approximation of the training data.

This perspective of placing uncertainty around the training data and resolving it by considering an adversarial evaluator leads to a zero-sum game formulation for inference and convex optimization for estimating model parameters.

The results on 130 prediction tasks showed that (1) the approach performs competitively with a state-of-the-art boosting method across many of these tasks and better on average. This is despite the fact that boosting, as an ensemble method, is able to implicitly consider a richer feature space for the classifiers that it ultimately produces. The performance of our approach is much more significantly better than structured multi-class SVM methods and reduction-based SVM methods, which are more directly comparable as they employ the same quadratic feature space.

3.3 Large Scale Game and Double Oracle

For problems like cost sensitive classification, the game matrix $\mathbf{C}_{\mathbf{X},\theta}$ has a size that is the number of classes, which is usually a small fix number. However, for some other problems, the $\mathbf{C}_{\mathbf{X},\theta}$ can be extremely large that directly using linear programming is not feasible.

We first encounter this problem in (2) where we want to solve the optimization problem of F-score and nDCG score.

For the problem of F-score, we can only calculate it out after we get the full prediction results for each of the the samples, so we can no longer treat each label independently, but view them altogether as a structured output. In this case, the output space is 2^n where n is the number of all the samples.

For problems that trying to maximizing a score instead of minimizing a lose, the (Equation 3.4) can be rewrite as

$$\max_{\theta} \mathbb{E}_{\mathbf{X},\mathbf{Y}\sim\tilde{p}} \left[\min_{\{\check{\mathbf{p}}_x\}\in\Delta} \max_{\{\hat{\mathbf{p}}_x\}\in\Delta} \hat{\mathbf{p}}_{\mathbf{X}}^{T}\, \mathbf{C}_{\mathbf{X},\theta}\, \check{\mathbf{p}}_{\mathbf{X}} \right] \tag{3.8}$$

For this problem, it can also be a hard problem using arbitrary feature functions. If we use an additive feature fucntion where $\Phi(y,x) = \sum_{i=1}^{n} \phi(x_i)\mathbf{1}_{y_i=1}(y_i)$, the game matrix for 3 sample classification problem is in Table I:

TABLE I: F-score for n=3 binary classification

F_1	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
000	1	$0-\psi_3$	$0-\psi_2$	$0-\psi_2-\psi_3$	$0-\psi_1$	$0-\psi_1-\psi_3$	$0-\psi_1-\psi_2$	$0-\psi_1-\psi_2-\psi_3$
001	0	$1-\psi_3$	$0-\psi_2$	$\frac{2}{3}-\psi_2-\psi_3$	$0-\psi_1$	$\frac{2}{3}-\psi_1-\psi_3$	$0-\psi_1-\psi_2$	$\frac{1}{2}-\psi_1-\psi_2-\psi_3$
010	0	$0-\psi_3$	$1-\psi_2$	$\frac{2}{3}-\psi_2-\psi_3$	$0-\psi_1$	$0-\psi_1-\psi_3$	$\frac{2}{3}-\psi_1-\psi_2$	$\frac{1}{2}-\psi_1-\psi_2-\psi_3$
011	0	$\frac{2}{3}-\psi_3$	$\frac{2}{3}-\psi_2$	$1-\psi_2-\psi_3$	$0-\psi_1$	$\frac{1}{2}-\psi_1-\psi_3$	$\frac{1}{2}-\psi_1-\psi_2$	$\frac{4}{5}-\psi_1-\psi_2-\psi_3$
100	0	$0-\psi_3$	$0-\psi_2$	$0-\psi_2-\psi_3$	$1-\psi_1$	$\frac{2}{3}-\psi_1-\psi_3$	$\frac{2}{3}-\psi_1-\psi_2$	$\frac{1}{2}-\psi_1-\psi_2-\psi_3$
101	0	$\frac{2}{3}-\psi_3$	$0-\psi_2$	$\frac{1}{2}-\psi_2-\psi_3$	$\frac{2}{3}-\psi_1$	$1-\psi_1-\psi_3$	$\frac{1}{2}-\psi_1-\psi_2$	$\frac{4}{5}-\psi_1-\psi_2-\psi_3$
110	0	$0-\psi_3$	$\frac{2}{3}-\psi_2$	$\frac{1}{2}-\psi_2-\psi_3$	$\frac{2}{3}-\psi_1$	$\frac{1}{2}-\psi_1-\psi_3$	$1-\psi_1-\psi_2$	$\frac{4}{5}-\psi_1-\psi_2-\psi_3$
111	0	$\frac{1}{2}-\psi_3$	$\frac{1}{2}-\psi_2$	$\frac{4}{5}-\psi_2-\psi_3$	$\frac{1}{2}-\psi_1$	$\frac{4}{5}-\psi_1-\psi_3$	$\frac{4}{5}-\psi_1-\psi_2$	$1-\psi_1-\psi_2-\psi_3$

This game matrix is exponential to the size of samples n. However, for many practical problems with very large game matrix, the equilibrium only contains strategies whose number is only polynomial to the the number of samples. As long as the number of equilibrium strategies is polynomial, there

is a method called double oracle (47) that can step by step search for the equilibrium starting from a small size game matrix. It operates by maintaining a set of label vectors $\hat{\mathcal{S}}$ and $\check{\mathcal{S}}$ for each player that it greedily expands until the game's equilibrium is supported by the label vectors of these sets. This is accomplished by repeatedly finding the game's equilibrium for the current set of strategies and then alternatively adding each player's best response against the other player's equilibrium distribution as a new row or column for the game. The algorithm (2) works as in algorithm 1.

Algorithm 1 Double Oracle Algorithm

Require: Lagrangian potentials ψ; initial action set $\hat{\mathcal{S}}_0$ and $\check{\mathcal{S}}_0$
Ensure: Nash equilibrium, $(\hat{P}, \check{P})$

Initialize player's action set $\hat{\mathcal{S}} \leftarrow \hat{\mathcal{S}}_0$; Initialize adversarial's actio set $\check{\mathcal{S}} \leftarrow \check{\mathcal{S}}_0$
$\mathbf{C}_{\mathbf{X},\theta} \leftarrow$ buildGameMatrix$(\hat{\mathcal{S}}, \check{\mathcal{S}}, \psi)$
repeat
  $\left[\hat{P}(\hat{y}|x), v_{Nash_1}\right] \leftarrow$ solveGame$(\mathbf{C}_{\mathbf{X},\theta})$
  $[\check{a}, \check{v}_{BR}] \leftarrow$ findBestResponceAction$(\hat{P}(\hat{y}|x), \psi)$
  if $v_{Nash_1} \neq \check{v}_{BR}$ **then**
    $\check{\mathcal{S}} \leftarrow \check{\mathcal{S}} \cup \check{a}$
    $\mathbf{C}_{\mathbf{X},\theta} \leftarrow$ buildGameMatrix$(\hat{\mathcal{S}}, \check{\mathcal{S}}, \psi)$
  end if
  $\left[\hat{P}(\hat{y}|x), v_{Nash_2}\right] \leftarrow$ solveGame$(\mathbf{C}_{\mathbf{X},\theta})$
  $[\hat{a}, \hat{v}_{BR}] \leftarrow$ findBestResponceAction$(\check{P}(\check{y}|x), \psi)$
  if $v_{Nash_2} \neq \hat{v}_{BR}$ **then**
    $\hat{\mathcal{S}} \leftarrow \hat{\mathcal{S}} \cup \hat{a}$
    $\mathbf{C}_{\mathbf{X},\theta} \leftarrow$ buildGameMatrix$(\hat{\mathcal{S}}, \check{\mathcal{S}}, \psi)$
  end if
until $v_{Nash_1} = v_{Nash_1} = \hat{v}_{BR} = \check{v}_{BR}$
return $(\hat{P}, \check{P})$

Neither player can improve upon their strategy with additional purestrategies when Algorithm 1 terminates, thus the mixed strategies it returns are a Nash equilibrium pair (47). Additionally, the algorithm is efficient in practice so long as each player's strategy is compact (i.e., the number of actions with non-zero probability is a polynomial subset of the label combinations) and best responses to opponents' strategies can be obtained efficiently (i.e., in polynomial time) for each player. Additionally, this algorithm can be modified to find approximate equilibria by limiting the number of actions for each player's set $\hat{\mathcal{S}}$ and $\check{\mathcal{S}}$.

The critical part to apply this algorithm is to find out the efficient algorithms to find the best response action of one player given a mixed strategy of the other player, and this is not easy for all the problems. For F-score and nDCG, we found the best responses (2). For some other problems like number of inversions, whether an efficient algorithm exist is still a question.

3.4 Fisher Consistency

Unlike SSVM and many surrogate hinge losses that do not have Fisher consistency, our adversarial approach keeps fisher consistency in most problems, including multiclass classification (28), graph models (48), oridinal regression (29) and bipartile matching (5).

In fact, our adversarial approach has very nice general consistency guarantees for most learning tasks (49).

For standard prediction that trying to maximize the potentials, i.e., $\text{argmax}_y f(x, y)$, where the predictor and the ground truth labels are chosen from the same set of labels, (Equation 2.4) is equivalent to

$$\mathbb{E}_{\mathbf{Y}|x\sim P}\left[\textbf{loss}(\text{argmax}_{y'} f^*(x, y'), \mathbf{Y})\right] = \min_f \mathbb{E}_{\mathbf{Y}|x\sim P}\left[\textbf{loss}(\text{argmax}_{y'} f(x, y'), \mathbf{Y})\right] \tag{3.9}$$

Since f is optimal over all measurable functions, it can be further simplified as:

$$\begin{aligned} & f^* \in \text{argmin}_f \mathbb{E}_{\mathbf{Y}|x\sim P}\left[\delta_f(x, \mathbf{Y})\right] \\ \Longleftarrow\ & \text{argmax}_{y'} f^*(x, y') \in \text{argmax}_{y'} \mathbb{E}_{\mathbf{Y}|x\sim P}\left[\text{loss}(y', \mathbf{Y})\right] \end{aligned} \tag{3.10}$$

where $\delta_f(x, y)$ is the surrogate loss function when true label equals to y. For our adversarial approach, the empirical prospective of it would be a empirical risk minimization problem with a surrogate loss function AL_{f_θ}:

$$\min_\theta \mathbb{E}_{\mathbf{Y}|x\sim \tilde{P}}\left[\text{AL}_{f_\theta}(\mathbf{X}, \mathbf{Y})\right] \tag{3.11}$$

$$\text{where } \text{AL}_f(x, y) = \min_{\hat{P}(\hat{Y}|x)} \max_{\check{P}(\check{Y}|x)} \mathbb{E}_{\substack{\hat{Y}|x\sim\hat{P} \\ \check{Y}|x\sim\check{P}}}\left[\text{loss}(\hat{Y}, \check{Y}) + f(x, \check{Y}) - f(x, y)\right] \tag{3.12}$$

We have following conclusion:

Theorem 2. *(49) In standard prediction setting, suppose loss$(y, y) <$ loss(y', y) for all $y' \neq y$. Then the adversarial permutation loss AL_f is Fisher consistent if f is optimized over all measurable functions and $\mathcal{Y}^\diamond$ is a singleton.*

here $\mathcal{Y}^\diamond$ is the set of loss minimizing labels. It is a singleton means the optimal label is unique.

Theorem 3. *(49) Suppose* $loss(y, y) < loss(y', y)$ *for all* $y' \neq y$. *Furthermore if* f *is over all measurable functions, then:*

(a) there exists $f^* \in \mathcal{F}^*$ *such that* $\operatorname{argmax}_\pi f^*(x, y) \subseteq \mathcal{Y}^\diamond$ *(i.e., satisfies the Fisher consistency requirement). In fact, all elements in* $\mathcal{Y}^\diamond$ *can be recovered by some* $f^* \in \mathcal{F}^*$.

(b) if $\operatorname{argmin}_y \sum_{y' \in \mathcal{Y}^\diamond} \alpha_{y'} loss(y', y) \subseteq \mathcal{Y}^\diamond$ *for all* $\alpha_{(\cdot)} \geq 0$; $\sum_{y' \in \mathcal{Y}^\diamond} \alpha_{y'} = 1$, *then* $\operatorname{argmax}_y f^*(x, y) \subseteq \mathcal{Y}^\diamond$ *for* ***all*** $f^* \in \mathcal{F}^*$. *In this case, all* $f^* \in \mathcal{F}^*$ *satisfy the Fisher consistency requirement.*

These assumptions of loss functions are very mild and for most classical learning task, they can be achieved.

In practice, we can proof the fisher consistency of adversarial learning for a particular problem using a easier specific approach. For example, for the multilabel prediction problem

Theorem 4. *Robust cut learning is Fisher consistent for the Hamming loss over all distributions.*

Proof. Optimizing over the feature space of all measurable positive pairwise functions and unrestricted unary functions allows us to optimize $\boldsymbol{\psi}$ directly subject to constraints on the pairwise potential. However, for the Hamming loss, those constraints are not relevant to the proof. Rewriting the optimization in terms of $\boldsymbol{\psi}$ functions, we have:

$$\min_{\boldsymbol{\psi}} \min_{\hat{P}(\hat{y}|x)} \max_{\check{P}(\check{y}|x)} \sum_{\hat{y},\check{y},y} \hat{P}(\hat{\mathbf{y}}|\mathbf{x}) \check{P}(\check{\mathbf{y}}|\mathbf{x}) P(y|\mathbf{x}) \left[\text{loss}(\hat{\mathbf{Y}}, \check{\mathbf{Y}}) \right.$$

$$\left. + \boldsymbol{\psi}(\check{y}, \mathbf{x}) - \boldsymbol{\psi}(y, \mathbf{x}) \right] \tag{3.13}$$

If $\hat{P}(\hat{y}_i|\mathbf{x}) \neq P(y_i|\mathbf{x})$ for any input $\mathbf{x}$, then the unary potential function correspond to y_i, $\boldsymbol{\psi}(y_i, \mathbf{x})$, can be chosen so that:

$$\sum_{\check{y}_i, y_i} \check{P}(\check{y}_i|\mathbf{x}) P(y_i|\mathbf{x}) \left(\boldsymbol{\psi}(\check{y}_i, \mathbf{x}) - \boldsymbol{\psi}(y_i, \mathbf{x})\right) = \tag{3.14}$$

$$\sum_{y_i} \left(\check{P}(y_i|\mathbf{x}) - P(y_i|\mathbf{x})\right) \boldsymbol{\psi}(y_i, \mathbf{x}) \to -\infty.$$

Thus, due solely to the unary portion of the constraints/potential function, $\check{P}(\check{y}_i|\mathbf{x}) = P(y_i|\mathbf{x})$. At the optima of Eq. (Equation 3.13), the potential terms will equate to zero, reducing to:

$$\min_{\hat{P}(\hat{\mathbf{y}}|\mathbf{x})} \max_{\check{P}(\check{\mathbf{y}}|\mathbf{x}) \in \Xi} \sum_{\hat{y}, \check{y}} \hat{P}(\hat{\mathbf{y}}|\mathbf{x}) \check{P}(\check{\mathbf{y}}|\mathbf{x}) \left(\text{loss}(\hat{y}, \check{y})\right), \tag{3.15}$$

$$\left(\text{where } \check{P}(\check{\mathbf{y}}|\mathbf{x}) \in \xi \Rightarrow \forall i, \forall y_i, \check{P}(\check{y}_i|\mathbf{x}) = P(y_i|\mathbf{x})\right)$$

$$= \min_{\hat{P}(\hat{\mathbf{y}}|\mathbf{x})} \max_{\check{P}(\check{\mathbf{y}}|\mathbf{x}) \in \xi} \sum_i \sum_{\hat{y}_i, \check{y}_i} \hat{P}(\hat{y}_i|\mathbf{x}) \check{P}(\check{y}_i|\mathbf{x}) (\hat{y}_i \neq \check{y}_i)$$

$$= \min_{\hat{P}(\hat{\mathbf{y}}|\mathbf{x})} \sum_i \sum_{\hat{y}_i, y_i} \hat{P}(\hat{y}_i|\mathbf{x}) P(y_i|\mathbf{x}) (\hat{y}_i \neq y_i)$$

which is exactly the Bayes risk minimizer for the Hamming loss. □

Thus, our adversarial approach overcomes a key theoretical limitation of maximum margin methods for binary multilabel prediction problem.

CHAPTER 4

DIRECT APPROXIMATION WITH DOUBLE ORACLE

(This chapter contains contents in paper "Arc: Adversarial robust cuts for semi-supervised and multi-label classification. (3)", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 1905-1907). It also contains contents in paper "Adversarial Learning for 3D Matching. (4)", In Conference on Uncertainty in Artificial Intelligence (pp. 869-878). PMLR. The copyright policy for author reusing of them can be found in Appendix A)

Many learning version of the intractable problems will contain the intractable problem as a subproblem in the inference or separation part. We will show how adversarial learning solving them through several particular examples.

4.1 Multiclass Classification with Pairwise Features

Comparing with classical classification problems, structured learning is used when the data features or labels are correlated under a varieties of structures. Even if a tiny correlation is introduced to make use of that information and train a classifier that can be very challenging. Multiclass classification with pairwise features is a seemingly simple problem that adds a pairwise feature between two objects, but we can show that it added the complexity of the models greatly and could be a NP-hard problem. And we try to solve it with the help of approximation methods.

4.1.1 Multiway Cuts

A multiway cut problem is defined as: Given a graph with weights assigned to the edges, and a set of terminals nodes, find out a set of edges that have the minimal summed weights such that after deleting this edges, there will be no path between any two terminal nodes.

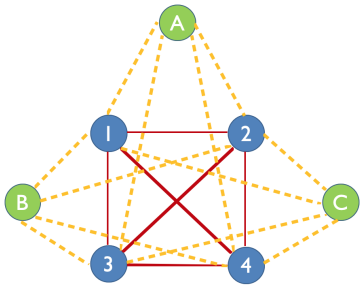

Figure 2: $\mathfrak{n} = 4$ Multiway cut problem with 4 internal nodes and 3 terminal nodes

There many approximation methods for solving the multi-cut problem (50), and linear programming relaxation (LPR) is one of the good and common methods (51; 52). In paper (10), they also found that LPR can achieve very good practical performance and had no fraction results.

4.1.2 Minimax Game Formulation

We introduce the distribution $\check{\mathrm{P}}(\check{\mathbf{Y}}|\mathbf{X})$ controlled by the adversary to produce worst-case approximations of the actual training example label, $\mathbf{y}^{(\mathrm{i})}$. The adversary seeks to maximize the loss subject

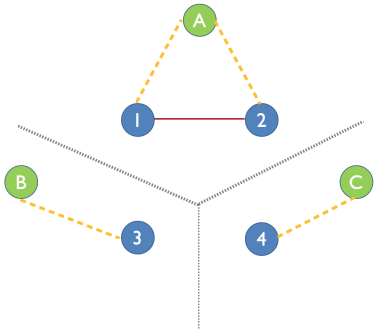

Figure 3: After removing the minimal cut

to certain constraints based on the training data sample, while the predictor player chooses $\hat{P}(\hat{\mathbf{Y}}|\mathbf{X})$ to minimize the expected loss:

$$\min_{\hat{P}(\hat{\mathbf{Y}}|\mathbf{X})} \max_{\check{P}(\check{\mathbf{Y}}|\mathbf{X})} \mathbb{E}_{\mathbf{X}\sim\tilde{P};\hat{\mathbf{Y}}|\mathbf{X}\sim\hat{P};\check{\mathbf{Y}}|\mathbf{X}\sim\check{P}} \left[\text{loss}(\hat{\mathbf{Y}},\check{\mathbf{Y}})\right] \text{ such that:}$$

$$\mathbb{E}_{\mathbf{X}\sim\tilde{P};\check{\mathbf{Y}}|\mathbf{X}\sim\check{P}} \left[\boldsymbol{\phi}_i(\check{Y}_i,\mathbf{X})\right] = \mathbb{E}_{\mathbf{X},\mathbf{Y}\sim\tilde{P}} \left[\boldsymbol{\phi}_i(Y_i,\mathbf{X})\right], \forall i \in [n];$$

$$\mathbb{E}_{\mathbf{X}\sim\tilde{P};\check{\mathbf{Y}}|\mathbf{X}\sim\check{P}} \left[\boldsymbol{\phi}_{i,j}(\check{Y}_i,\check{Y}_j,\mathbf{X})\right] = \mathbb{E}_{\mathbf{X},\mathbf{Y}\sim\tilde{P}} \left[\boldsymbol{\phi}_{i,j}(Y_i,Y_j,\mathbf{X})\right], \forall i,j \in [n], \tag{4.1}$$

where $\tilde{P}$ is the empirical distribution of $\mathbf{X}$ [and $\mathbf{Y}$] in the training data set. $\boldsymbol{\phi}_i(Y_i,X)$ are unary features and $\boldsymbol{\phi}_{i,j}(Y_i,Y_j,X)$ are pairwise features. For multiclass classification, the loss we use is simply the Hamming loss

$$\text{loss}(\hat{\mathbf{y}},\check{\mathbf{y}}) = \frac{1}{n}\sum_i \mathbf{1}_{\hat{y}_i=\check{y}_i}(\hat{y}_i,\check{y}_i), \tag{4.2}$$

Conceptually, the adversary seeks to construct a label distribution that is as uncertain as possible, since this forces the predictor to incur large amounts of expected loss. However, the constraints placed on the adversary to match statistics ϕ relating the actual training data labels to inputs $\mathbf{x}$ and one another (such as mean or higher order moments) prevent the adversary from doing this, and, when chosen carefully, can restrict the adversary to be highly predictable.

We reduce and generalize the restrictions on the adversary by redefining the constraints so that they share clique features. Rather than having unique pairwise constraints for each $i \neq j$, we can have a single constraint:

$$\mathbb{E}_{\mathbf{X}\sim\tilde{P},\check{\mathbf{Y}}|\mathbf{X}\sim\check{P}}\left[\sum_{i\neq j}\boldsymbol{\phi}_{i,j}(\check{Y}_i,\check{Y}_j,\mathbf{X})\right]=\mathbb{E}_{\mathbf{X},\mathbf{Y}\sim\tilde{P}}\left[\sum_{i\neq j}\boldsymbol{\phi}_{i,j}(Y_i,Y_j,\mathbf{X})\right]. \tag{4.3}$$

Unlike the binary classification case (3), there is no efficient algorithm that requires a non negative pairwise feature. So we can directly require the advertorial's expected pairwise feature equal to the expected pairwise feature got from training sample.

Using the method of Lagrange multipliers and strong duality (53), the constraints of this formulation can be incorporated as Lagrangian potentials with parameters $\{\theta_i\}$ and $\{\theta_{i,j}\}$:

$$\begin{aligned}\min_{\{\theta_i\},\{\theta_{i,j}\}}\min_{\hat{P}(\hat{\mathbf{y}}|\mathbf{x})}\max_{\check{P}(\check{\mathbf{y}}|\mathbf{x})}\mathbb{E}_{\mathbf{X},\mathbf{Y}\sim\tilde{P},\hat{\mathbf{Y}}|\mathbf{X}\sim\hat{P},\check{\mathbf{Y}}|\mathbf{X}\sim\check{P}}\Big[&\mathrm{loss}(\hat{\mathbf{Y}},\check{\mathbf{Y}})\\&+\sum_i\theta_i\cdot(\boldsymbol{\phi}_i(\check{Y}_i,\mathbf{X})-\boldsymbol{\phi}_i(Y,\mathbf{X}))\\&+\sum_{i\neq j}\theta_{i,j}\cdot(\boldsymbol{\phi}_{i,j}(\check{Y}_i,\check{Y}_j,\mathbf{X})-\boldsymbol{\phi}_{i,j}(Y_i,Y_j,\mathbf{X}))\Big].\end{aligned} \tag{4.4}$$

These Lagrangian potentials exactly match those of the Markov random field (Equation 2.2) in form, and can similarly be written as $\Psi(\check{\mathbf{y}}, \mathbf{x}) = \sum_i \psi_i(y_i, \mathbf{x}) + \sum_{i \neq j} \psi_{i,j}(y_i, y_j, \mathbf{x})$, where $\psi_i(y_i, \mathbf{x}) = \theta_i \cdot \phi_i(y_i, \mathbf{x})$ and $\psi_{i,j}(y_i, y_j, \mathbf{x}) = \theta_{i,j} \cdot \phi_i(y_i, y_j, \mathbf{x})$.

Under the Hamming loss (Equation 4.2), the terms of this optimization problem in Equation (Equation 4.4) involving $\hat{P}$ and $\check{P}$ for a specific input $\mathbf{x}$ can be re-written as a bilinear function of the predictor's distribution, the Lagrangian-augmented loss function, and the adversary's distribution,

$$\underbrace{\begin{bmatrix} \hat{P}(000) \\ \hat{P}(001) \\ \hat{P}(002) \\ \hat{P}(010) \\ \hat{P}(011) \\ \vdots \end{bmatrix}^{T}}_{\hat{\mathbf{p}}_{\mathbf{x}}} \underbrace{\begin{bmatrix} 0 + \Psi(000, \mathbf{x}) & \frac{1}{3} + \Psi(001, \mathbf{x}) & \cdots \\ \frac{1}{3} + \Psi(000, \mathbf{x}) & 0 + \Psi(001, \mathbf{x}) & \cdots \\ \frac{1}{3} + \Psi(000, \mathbf{x}) & \frac{1}{3} + \Psi(001, \mathbf{x}) & \cdots \\ \frac{1}{3} + \Psi(000, \mathbf{x}) & \frac{2}{3} + \Psi(001, \mathbf{x}) & \cdots \\ \frac{2}{3} + \Psi(000, \mathbf{x}) & \frac{1}{3} + \Psi(001, \mathbf{x}) & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}}_{\mathbf{C}_{\theta,\mathbf{x}} = \{\text{loss}(\hat{\mathbf{y}}, \check{\mathbf{y}}) + \Psi(\check{\mathbf{y}}, \mathbf{x})\}} \underbrace{\begin{bmatrix} \check{P}(000) \\ \check{P}(001) \\ \check{P}(002) \\ \check{P}(010) \\ \check{P}(011) \\ \vdots \end{bmatrix}}_{\check{\mathbf{p}}_{\mathbf{x}}},$$

which can then be compactly denoted in terms of vectors of conditional probabilities ($\hat{\mathbf{p}}_{\mathbf{x}}$ and $\check{\mathbf{p}}_{\mathbf{x}}$), and a payoff matrix ($\mathbf{C}_{\theta,\mathbf{x}}$) for each example $\mathbf{x}$.

With the Lagrangian potentials incorporated into the payoff matrix $\mathbf{C}_{\theta,\mathbf{x}}$, the joint game over all training instances can be solved independently for each one:

$$\min_{\{\theta_i\},\{\theta_{i,j}\} \leq 0} \mathbb{E}_{\mathbf{X},\mathbf{Y} \sim \tilde{P}} \left[\left(\max_{\check{\mathbf{p}}_{\mathbf{X}}} \min_{\hat{\mathbf{p}}_{\mathbf{X}}} \hat{\mathbf{p}}_{\mathbf{X}}^{T} \, \mathbf{C}_{\theta_1,\mathbf{X}} \, \check{\mathbf{p}}_{\mathbf{X}} \right) - \sum_i \theta_i \cdot \boldsymbol{\phi}_i(Y_i, \mathbf{X}) - \sum_{i \neq j} \theta_{i,j} \cdot \boldsymbol{\phi}_{i,j}(Y_i, Y_j, \mathbf{X}) \right]. \tag{4.5}$$

Unfortunately, naïvely constructing the game in this manner requires time and space that grow exponentially with the number of predicted variables to include every possible vector of values, as the game matrix sized is $(k^n)^2$. We address this computational bottleneck using the constraint generation methods in a later sections.

4.1.3 Double Oracle

Like that in ARC (3), this approach obtains an equilibrium over value assignments for the set of predicted variables between the predictor and the adversary, which means this game is played over the set of all possible label assignments as strategies. And it also have the property that in practice the game equilibria have sparse support (i.e., $P(\mathbf{y}|\mathbf{x}) = 0$ for most $\mathbf{y}$). We can still employ the double oracle algorithm (47) to uncover a set of strategies for each player that support the equilibrium.

The algorithm 2 used in (3) can work also here with minor modification.

Within the algorithm, we denote the potentials for the label vectors of $\check{\mathcal{S}}$ as $\boldsymbol{\Psi}(\check{\mathcal{S}}) \triangleq \{\Psi(\check{\mathbf{y}}, \mathbf{x})\}_{\check{\mathbf{y}} \in \check{\mathcal{S}}}$ and denote the loss matrix between all pairs of label vectors across sets as

$$\text{loss}_{\text{Ham}}(\hat{\mathcal{S}}, \check{\mathcal{S}}) \triangleq \{\text{HammingLoss}(\check{\mathbf{y}}, \hat{\mathbf{y}})\}_{\check{\mathbf{y}} \in \check{\mathcal{S}}, \hat{\mathbf{y}} \in \hat{\mathcal{S}}} \tag{4.6}$$

We solve the zero-sum games (solveGame) as linear programs (54), which can be solved in polynomial time. The key remaining challenge is finding the best response for each player.

Algorithm 2 Double Oracle Algorithm for MCPF Equilibria

Require: Node features $\{\phi_i(\cdot, \mathbf{x})\}$; Pairwise features $\{\phi_{i,j}(\cdot, \cdot, \mathbf{x})\}$; Parameters θ; Initial label $\mathbf{y}_{\text{initial}}$
Ensure: The (sparse) Nash equilibrium strategies sets: $\check{\mathcal{S}}$,$\hat{\mathcal{S}}$
The (sparse) Nash equilibrium strategies distributions: $\hat{P}$,$\check{P}$
The (sparse) Nash equilibrium values: $\hat{V}$,$\check{V}$

$\check{\mathcal{S}} \leftarrow \hat{\mathcal{S}} \leftarrow \{\mathbf{y}_{\text{initial}}\}$
repeat
  $(\hat{P}, \check{P}, \check{V}) \leftarrow \text{solveGame}(\boldsymbol{\Psi}(\check{\mathcal{S}}), \text{loss}_{\text{Ham}}(\check{\mathcal{S}}, \hat{\mathcal{S}}))$
  $(\check{\mathbf{y}}_{new}, V_{\max}) \leftarrow \max_{\check{\mathbf{y}}} \mathbb{E}_{\hat{\mathbf{Y}} \sim \hat{P}}[\text{loss}_{\text{Ham}}(\check{\mathbf{y}}, \hat{\mathbf{Y}}) + \Psi(\check{\mathbf{y}})]$
  if $(\check{V} \neq V_{\max})$ **then**
    $\check{\mathcal{S}} \leftarrow \check{\mathcal{S}} \cup \check{\mathbf{y}}_{new}$
  end if
  $(\hat{P}, \check{P}, \hat{V}) \leftarrow \text{solveGame}(\boldsymbol{\Psi}(\check{\mathcal{S}}), \text{loss}_{\text{Ham}}(\check{\mathcal{S}}, \hat{\mathcal{S}}))$
  $(\hat{\mathbf{y}}_{new}, V_{\min}) \leftarrow \min_{\hat{\mathbf{y}}} \mathbb{E}_{\check{\mathbf{Y}} \sim \check{P}}[\text{loss}_{\text{Ham}}(\check{\mathbf{Y}}, \hat{\mathbf{y}})]$
  if $(\hat{V} \neq V_{\min})$ **then**
    $\hat{\mathcal{S}} \leftarrow \hat{\mathcal{S}} \cup \hat{\mathbf{y}}_{new}$
  end if
until $\check{V} = V_{\max} = \hat{V} = V_{\min}$
return $(\hat{P}, \check{P})$

4.1.3.1 Adversary's best responses:

Given the predictor's distribution over value vectors, $\hat{P}(\hat{\mathbf{y}}|\mathbf{x})$, over $\hat{\mathbf{y}} \in \hat{\mathcal{S}}$ the adversary's best response is a single vector of values $\check{\mathbf{y}}_{\text{new}}$ with the largest possible expected value under $\hat{P}$. It is determined both by the expected loss against the predictor's distribution and the Lagrangian potential terms:

$$\underset{\check{\mathbf{y}}}{\text{argmax}}\ \mathbb{E}_{\hat{\mathbf{Y}}|\mathbf{x} \sim \hat{P}} \left[\sum_i \mathbf{1}_{\hat{Y}_i \neq \check{y}_i}(\hat{Y}_i, \check{y}_i) \right] + \sum_i \psi_i(\check{\mathbf{y}}_i, \mathbf{x}) + \sum_{i,j} \psi_{i,j}(\check{\mathbf{y}}_i, \check{\mathbf{y}}_j, \mathbf{x}). \tag{4.7}$$

To solve Eq. (Equation 4.7), we introduce indicator variables $u_{i,k}$ and $u_{i,j}^{(k)} \in \{0,1\}$. $u_{i,k}$ is 1 only if $y_i = k$, and $u_{i,j}^{(k)}$ is 1 only if $y_i = y_j = k$ where $k \in \mathcal{Y}$ is one of the possible labels. Combining them all into a single vector $\mathbf{u}$ of size q, the problem (Equation 4.7) can be formulated as an integer linear programming (ILP) problem:

$$\begin{aligned}
&\text{argmax}_{\mathbf{u}\in\{0,1\}^q} \sum_i \sum_k u_{i,k} \cdot \alpha_i(k, x_i) + \sum_{i,j} \sum_{k\in\mathcal{Y}} u_{i,j}^{(k)} \cdot \alpha_{i,j}(\check{y}_i, \check{y}_j, x_i, x_j) \\
&\text{such that: } \sum_k u_{i,k} = 1, \forall i \in [n] \\
&\qquad u_{i,j}^{(k)} \leq u_{i,k}, \forall i, j \in [n], k \in \mathcal{Y} \\
&\qquad u_{i,j}^{(k)} \leq u_{j,k}, \forall i, j \in [n], k \in \mathcal{Y}
\end{aligned} \tag{4.8}$$

where α_i and $\alpha_{i,j}$ are defined as:

$$\alpha_i(k, \mathbf{x}) = \frac{1}{n} \sum_{\hat{\mathbf{y}}} \hat{P}(\hat{\mathbf{y}}|\mathbf{x}) \mathbf{1}_{\hat{y}\neq k}(\hat{y}, k) + \boldsymbol{\psi}_i(k, \mathbf{x}), \forall i \in [n] \tag{4.9}$$

$$\alpha_{i,j}(\check{\mathbf{y}}_i, \check{\mathbf{y}}_j, \mathbf{x}) = \sum_k \left(\mathbf{1}_{y_i=k}(y_i, k) \cdot \mathbf{1}_{y_j=k}(y_j, k)\right) \theta_{i,j} \cdot \boldsymbol{\sigma}_{i,j}(k, \mathbf{x}), \forall i, j \in [n]. \tag{4.10}$$

Here $u_{i,j}^{(k)} \leq u_{j,k}$ comes from $u_{i,j}^{(k)} = u_{i,k} \cdot u_{j,k}$. Here we construct the pairwise features as non zero only when those two nodes have the same label. So $\boldsymbol{\phi}_{i,j}(y_i, y_j, \mathbf{x})$ can further be simplified as $\boldsymbol{\sigma}_{i,j}(k, \mathbf{x})$.

Eq. (Equation 4.8) can be viewed as a multiway cut problem by:

1. Treat every samples as internal node i.
2. Add k additional terminal nodes k.

3. Use $\alpha_i(k, \mathbf{x})$ as edge weights between node $\mathbf{x}$) and node k.

4. Use $\alpha_{i,j}(\check{\mathbf{y}}_i, \check{\mathbf{y}}_j, \mathbf{x})$ as edge weights between node i and j

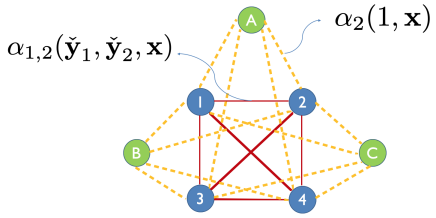

Figure 4: $n = 4$ Multiway cut problem with 4 samples and 3 classes

Then Eq.(Equation 4.8) means the maximal summed weight of all possible combinations of the edges in the graph we construct. One simple observation of why this is a multiway cut problem is looking at the min-cut in the opposite way. Min-cut is deleting minimal weighted edges from the original graph to make terminal nodes disconnected, which is equivalent to add edges from a node only graph. You can add edges as much as you can as long as the terminal nodes are not connected. Then when the maximal of Eq. (Equation 4.8) is achieved, every node i must and can only connect to one of the terminal nodes. For the edges connect between node i, it is non-zero (connected) only if they have the same label, which means they are connected to the same terminal node k. So by solving Eq. (Equation 4.8), there cannot be any path between the k additional nodes. And every connected subgraph matches a class in the original problem.

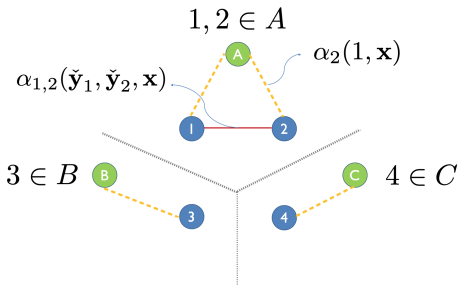

Figure 5: After removing the minimal cut

Actually, (Equation 4.8) can be approximated by Linear Programming relaxation directly instead of converging it into a multiway cut problem. Many works have shown that using LPR as separation oracle in learning methods can both theoretically and practically have a very good performance (55; 15).

4.1.3.2 Predictor's best response:

Given the adversary's distribution of value vectors $\check{P}(\check{\mathbf{y}}|\mathbf{x})$ from $\check{\mathbf{y}} \in \check{\mathcal{S}}$, the predictor's best response is a single vector of values with the smallest expected value against $\check{P}$. Since the Lagrangian potential does not rely on predictor's strategy, and the Hamming loss is additively decomposable, same as that in ARC paper (3). Because we have:

$$\begin{aligned} &\min_{\hat{\mathbf{y}}} \mathbb{E}_{\check{\mathbf{Y}}|\mathbf{x}\sim\check{P}} \left[\frac{1}{n} \sum_i \mathbf{1}_{\hat{y}_i \neq \check{Y}_i}(\hat{y}_i, \check{Y}_i) \right] \\ &= \frac{1}{n} \sum_i \min_{\hat{y}_i} \sum_{\check{\mathbf{y}}} \check{P}(\check{\mathbf{y}}|\mathbf{x}) \mathbf{1}_{\hat{y}_i \neq \check{y}_i}(\hat{y}_i, \check{y}_i). \end{aligned} \tag{4.11}$$

Thus, the predictor's best responses can be independently made for each variable:

$$\hat{y}_i = \underset{y}{\operatorname{argmax}}\, \check{P}(\check{Y}_i = y|\mathbf{x}). \tag{4.12}$$

Using these best response subroutines, the double oracle algorithm can be applied to obtain a compact equilibrium distribution for the robust min-cut game.

4.1.4 Parameter Learning

The final aspect of our approach is optimizing the model parameters θ_i and $\theta_{i,j}$, that incentivize the adversary. We accomplish this using standard tools from convex optimization. Specifically, we employ AdaGrad (56) and compute the gradients $\mathbf{g}_i$ and $\mathbf{g}_{i,j}$ for a single example $(\mathbf{x}^{(k)}, \mathbf{y}^{(k)})$ as the difference between the expected features under the adversary's distribution and the empirical features calculated from the training data:

$$\left\{\mathbb{E}_{\check{\mathbf{Y}}|\mathbf{x}^{(k)}\sim\check{P}}[\boldsymbol{\phi}_i(\check{Y}_i, \mathbf{x}^{(k)})] - \boldsymbol{\phi}_i(y_i^{(k)}, \mathbf{x}^{(k)})\right\} \text{ and} \tag{4.13}$$

$$\left\{\mathbb{E}_{\check{\mathbf{Y}}|\mathbf{x}^{(k)}\sim\check{P}}[\boldsymbol{\phi}_{i,j}(\check{Y}_i, \check{Y}_j, \mathbf{x}^{(k)})] - \boldsymbol{\phi}_{i,j}(y_i^{(k)}, y_j^{(k)}, \mathbf{x}^{(k)})\right\}. \tag{4.14}$$

4.1.5 Prediction

After get the predictor's distribution $\hat{P}$ and adversary's distribution $\check{P}$ from the equilibrium, we need to give the real prediction to our problem. There are many ways to do it and the simplest one is pick the most possible strategy (label vector) from the predictor's strategy distribution. However, since we

usually will get a mixture of strategies and the one with highest probability among them may not be the best single strategy. Usually the strategy got from

$$\min_{\hat{y}} \max_{\check{y}} \left[\text{loss}(\hat{y}, \check{y}) + \sum_{i} \Psi(\check{y}, x) \right] \tag{4.15}$$

would give better performance. However, since our game matrix is exponentially large, we can instead try to calculate:

$$\min_{\hat{y} | \hat{P}(\hat{y}) \neq 0} \max_{\check{y}} \left[\text{loss}(\hat{y}, \check{y}) + \sum_{i} \Psi(\check{y}, x) \right] \tag{4.16}$$

In which the inner problem will be the adversary's best response given a single predictor's strategy. Since we expect the equilibrium of the game will be polynomial size, we just need to repeat it polynomial times.

We should note that this can be done only after we get the equilibrium. And to get the equilibrium, we just need to put the test sample x inside the inner game Eq. (Equation 3.6) and solve it by double oracle with the trained θ.

4.1.6 Fisher Consistency

Since in this model we used an approximation method instead of exact value, previous theorems for the Fisher consistency of adversarial method no longer hold. Heuristically, we expect adversarial methods could also achieve better results, since our approach always trying to compete with worst case adversaries, it should be robust to small errors introduced by approximation methods. A concrete theoretical proof will be one of our future work.

4.2 3 Dimensional Matching

4.2.1 Weighted Maximum 3D Matching Problem

Given three sets of elements A , B and C of equal size ($|A| = |B| = |C| = n$), a perfect 3D matching $\pi \subseteq A \times B \times C$ contains one-to-one-to-one mappings (a, b, c) where $a \in A$, $b \in B$ and $c \in C$. π also needs to satisfy the conditions that: 1) o mappings share the same element; and 2) Elements in the mappings cover $A \cup B \cup C$.

An example of this problem is n people picking n different pieces of equipment to do n different tasks. Figure Figure 6 provides a visual representation of a 3D matching task. A matching π can be represented as a pair of permutations (π_1, π_2) where $(\pi_{1,i}, \pi_{2,i}), i \in [n] = 1, \cdots, n$ means the the ith element in the A match with the $\pi_{1,i}$th element in B and $\pi_{2,i}$th element in C. Suppose that a perfect matching has the additive potential $\psi(\pi_1, \pi_2) = \sum_i \psi_i(\pi_{1,i}, \pi_{2,i})$ The problem of maximum weighted 3D matching is to find π^* that maximizes this value. The set of possible solutions Π is simply all pairs of permutations with n elements.

Unlike the 2D setting, in which inference reduces to weighted maximum bipartite matching, a well studied problem with polynomial time solutions, 3D matching is NP-hard (generalizing the NP-hard 0-1 version (32)) (32).

We synthesize a 3D matching problem from a three-frame video tracking problem. To do so, we use the ground truth bounding boxes in the data as known objects and focus on matching them across frames. In contrast with existing tracking methods, which only consider relationships between consecutive frames, we also incorporate relationships between the first and third frame. Though somewhat

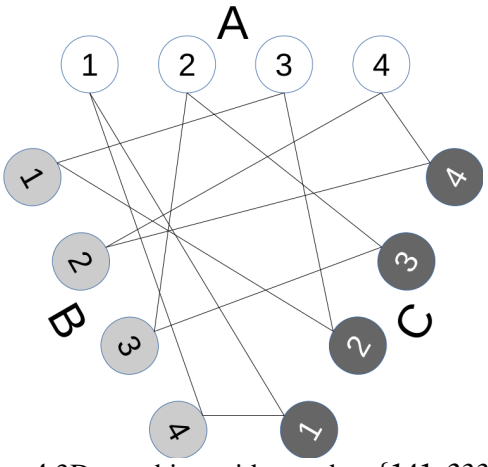

Figure 6: $n = 4$ 3D matching with matches $\{141, 233, 312, 424\}$

artificial for this domain, this prevents existing successful methods based on min-cost flow network algorithms (57; 58; 59; 60; 61) from being applicable.

The gap in hardness between bipartite matching and 3D matching is substantial, yet due to the similar form of the problem, the solution of adversarial bipartite matching provides a new a approach to efficiently solve the learning version of the 3D matching problem. Instead of solving 3D matching directly, we relax the problem into the space of marginal distributions and solve it under the framework of adversarial learning, leaving the hardest problem to the prediction stage. This technique opens a window for solving additional hard problems in similar way. Our major contributions are: 1) Adversarially formulating the 3D matching learning and prediction problem. 2) Providing detailed solutions for solving the relaxed optimization problem in the learning phase. 3) Demonstrating the effectiveness of adversarial learning on hard learning tasks.

4.2.2 Minimax Game Formulation

In 3D matching problem, the predicted label y is 2 sequence of permutations, and y_i are highly correlated. To make the problem more clear, we explicitly rewrite y as a pair (π_1, π_2) where $\pi_{1,i}$ is the entry in the second column that matchs ith entry in the first column and $\pi_{2,i}$ is the entry in the third column that matchs ith entry in the first column, then Π_1 and Π_2 are their corresponding random variables. We also denote $\phi(\pi_1, \pi_2, x)$ as a vector that enumerates the joint feature representations based on the 3D graph x and the matching assignment π_1 and π_2. This joint feature is defined additively over each node assignment, i.e., $\phi(\pi_1, \pi_2, x) = \sum_{i=1}^{n} \phi_i(\pi_{1,i}, \pi_{2,i}, x)$.

Following the adversarial approach applied to the task of learning bipartite matchings/permutations (5), we can build the following objective function for the task of learning 3D matchings:

$$\min_{\hat{P}(\hat{\Pi}_1,\hat{\Pi}_2|\mathbf{X})} \max_{\check{P}(\check{\Pi}_1,\check{\Pi}_2|\mathbf{X})} \mathbb{E}_{\mathbf{X}\sim\tilde{P};\hat{\Pi}_1,\hat{\Pi}_2|\mathbf{X}\sim\hat{P};\check{\Pi}_1,\check{\Pi}_2|\mathbf{X}\sim\check{P}} \left[\text{loss}(\hat{\Pi}_1, \hat{\Pi}_2, \check{\Pi}_1, \check{\Pi}_2)\right] \text{ s.t.} \tag{4.17}$$
$$\mathbb{E}_{\mathbf{X}\sim\tilde{P},\check{\Pi}_1,\check{\Pi}_2|\mathbf{X}\sim\check{P}} \left[\sum_{i=1}^{n} \phi_i(\check{\Pi}_{1,i}, \check{\Pi}_{2,i}, \mathbf{X})\right] = \mathbb{E}_{(\mathbf{X},\Pi_1,\Pi_2)\sim\tilde{P}} \left[\sum_{i=1}^{n} \phi_i(\Pi_{1,i}, \Pi_{2,i}, \mathbf{X})\right].$$

Here, Π_1 and Π_2 are the random variables of the permutation of the elements in the second and the third set. The constraint Ξ simply forces the adversary to produce mean feature values that match with the mean feature values of the training data sample. Applying the method of Lagrangian multipliers and strong duality for convex-concave saddle point problems (46; 62), the optimization in Eq. (Equation 4.17) can be equivalently solved in the dual formulation:

$$\min_{\theta} \mathbb{E}_{\mathbf{x},\Pi_1,\Pi_2\sim\tilde{P}} \min_{\hat{P}(\hat{\Pi}_1,\hat{\Pi}_2|\mathbf{x})} \max_{\check{P}(\check{\Pi}_1,\check{\Pi}_2|\mathbf{x})} \mathbb{E}_{\substack{\hat{\Pi}_1,\hat{\Pi}_2|\mathbf{x}\sim\hat{P} \\ \check{\Pi}_1,\check{\Pi}_2|\mathbf{x}\sim\check{P}}} \Big[\text{loss}(\hat{\Pi}_1,\hat{\Pi}_2,\check{\Pi}_1,\check{\Pi}_2) + \theta \cdot \sum_{i=1}^{n} \left(\phi_i(\check{\Pi}_{1,i},\check{\Pi}_{2,i},\mathbf{x}) - \phi_i(\Pi_{1,i},\Pi_{2,i},\mathbf{x}) \right) \Big], \tag{4.18}$$

where θ is the Lagrange dual variable for the moment matching constraints. For this problem, we use the match-based Hamming distance,

$$\text{loss}(\hat{\pi}_1,\hat{\pi}_2,\check{\pi}_1,\check{\pi}_2) = \frac{1}{n} \sum_{i=1}^{n} 1_{\hat{\pi}_{1,i}\neq\check{\pi}_{1,i}}(\hat{\pi}_{1,i},\check{\pi}_{1,i}) \vee 1_{\hat{\pi}_{2,i}\neq\check{\pi}_{2,i}}(\hat{\pi}_{2,i},\check{\pi}_{2,i}) \tag{4.19}$$

as the loss function. It means that all the three elements in a mapping must match with the ground truth, otherwise a loss of one is produced.

Let's define vector $\check{\mathbf{p}}_{\mathbf{X}}, \hat{\mathbf{p}}_{\mathbf{X}} \in \Delta$, which is the vector of conditional probabilities of all the possible $\hat{y}$ or $\check{y}$ given x. Then we can rewrite (Equation 4.18) in matrix form as:

$$\min_{\theta} \mathbb{E}_{\mathbf{x},\Pi_1,\Pi_2\sim\tilde{P}} \left[\max_{\check{\mathbf{p}}_{\mathbf{X}}} \min_{\hat{\mathbf{p}}_{\mathbf{X}}} \hat{\mathbf{p}}_{\mathbf{X}}^{T} \, \mathbf{C}_{\mathbf{X},\theta} \, \check{\mathbf{p}}_{\mathbf{X}} - \sum_{i} \theta \cdot \boldsymbol{\phi}(\Pi_{1,i},\Pi_{2,i},\mathbf{X}) \right]. \tag{4.20}$$

Table II is the payoff matrix $\mathbf{C}_{\theta,\mathbf{X}}$ for the game of size $n = 3$ with $3!^2$ actions (permutations) for the predictor player $\hat{\pi}$ and also for the adversarial approximation player $\check{\pi}$. Here, we define the difference between the Lagrangian potential of the adversary's action and the ground truth permutation as $\delta_{\check{\pi}_1,\check{\pi}_2} = \psi(\check{\pi}_1,\check{\pi}_2) - \psi(\pi_1,\pi_2) = \theta \cdot \sum_{i=1}^{n} \left(\phi_i(\check{\pi}_{i,1},\check{\pi}_{i,2},x) - \phi_i(\pi_{i,1},\pi_{i,2},x) \right)$.

TABLE II: Augmented Hamming loss matrix for n=3 permutations.

	$123,123$	$123,132$	$123,213$	$123,231$	$123,312$	$123,321$	$\cdots$
$123,123$	$0+\delta_{123,123}$	$\frac{2}{3}+\delta_{123,132}$	$\frac{2}{3}+\delta_{123,213}$	$1+\delta_{123,231}$	$1+\delta_{123,312}$	$\frac{2}{3}+\delta_{123,321}$	$\cdots$
$123,132$	$\frac{2}{3}+\delta_{123,123}$	$0+\delta_{123,132}$	$1+\delta_{123,213}$	$\frac{2}{3}+\delta_{123,231}$	$\frac{2}{3}+\delta_{123,312}$	$1+\delta_{123,321}$	$\cdots$
$123,213$	$\frac{2}{3}+\delta_{123,123}$	$1+\delta_{123,132}$	$0+\delta_{123,213}$	$\frac{2}{3}+\delta_{123,231}$	$\frac{2}{3}+\delta_{123,312}$	$1+\delta_{123,321}$	$\cdots$
$123,231$	$1+\delta_{123,123}$	$\frac{2}{3}+\delta_{123,132}$	$\frac{2}{3}+\delta_{123,213}$	$0+\delta_{123,231}$	$1+\delta_{123,312}$	$\frac{2}{3}+\delta_{123,321}$	$\cdots$
$123,312$	$1+\delta_{123,123}$	$\frac{2}{3}+\delta_{123,132}$	$\frac{2}{3}+\delta_{123,213}$	$1+\delta_{123,231}$	$0+\delta_{123,312}$	$\frac{2}{3}+\delta_{123,321}$	$\cdots$
$123,321$	$\frac{2}{3}+\delta_{123,123}$	$1+\delta_{123,132}$	$1+\delta_{123,213}$	$\frac{2}{3}+\delta_{123,231}$	$\frac{2}{3}+\delta_{123,312}$	$0+\delta_{123,321}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\ddots$

Since the number of permutations (π_1,π_2) is $(\mathcal{O}(n!^2))$, we are not able to solve the game directly even for fairly small n.

4.2.3 Double Oracle

We can try double oracle algorithm 3 again to solve the inner problem, first

$$\max_{\check{\mathbf{p}}_{\mathbf{X}}}\min_{\hat{\mathbf{p}}_{\mathbf{X}}}\hat{\mathbf{p}}_{\mathbf{X}}^{\mathsf{T}}\,\mathbf{C}_{\mathbf{X}}\,\check{\mathbf{p}}_{\mathbf{X}}-\sum_{i}\theta\cdot\boldsymbol{\phi}(\Pi_{1,i},\Pi_{2,i},\mathbf{X}) \tag{4.21}$$

and use gradient decent method to get θ after it.

4.2.3.1 Predictor's best response:

Let's explore its solution starting from the easier part–predictor's best response. Given the current adversary's distribution of strategies $\check{P}$ (assuming there are l strategies in it) in the double oracle process,

Algorithm 3 Double Oracle Algorithm for 3 Dimensional Matching Equilibrium.

Require: Lagrangian potentials $\Psi(\cdot)$; Initial Matching $\mathbf{Y}_{\text{initial}}$
Ensure: The (sparse) Nash equilibrium strategies sets: $\check{\mathcal{S}}$,$\hat{\mathcal{S}}$
The (sparse) Nash equilibrium strategies distributions: $\hat{\mathsf{P}}$,$\check{\mathsf{P}}$
The (sparse) Nash equilibrium values: $\hat{\mathsf{V}}$,$\check{\mathsf{V}}$

$\check{\mathcal{S}} \leftarrow \hat{\mathcal{S}} \leftarrow \{\mathbf{Y}_{\text{initial}}\}$
repeat
  $(\hat{\mathsf{P}}, \check{\mathsf{P}}, \check{\mathsf{V}}) \leftarrow \text{solveGame}(\boldsymbol{\Psi}(\check{\mathcal{S}}), \text{loss}(\hat{\mathcal{S}}, \check{\mathcal{S}}))$
  $(\check{\mathbf{Y}}_{\text{new}}, \mathsf{V}_{\max}) \leftarrow \text{argmax}_{\check{\mathbf{Y}}} \mathbb{E}_{\hat{\mathbf{Y}} \sim \hat{\mathsf{P}}}[\text{loss}(\hat{\mathbf{Y}}, \check{\mathbf{Y}}) + \Psi(\check{\mathbf{Y}})]$
  if $(\check{\mathsf{V}} \neq \mathsf{V}_{\max})$ **then** $\check{\mathcal{S}} \leftarrow \check{\mathcal{S}} \cup \check{\mathbf{Y}}_{\text{new}}$
  $(\hat{\mathsf{P}}, \check{\mathsf{P}}, \hat{\mathsf{V}}) \leftarrow \text{solveGame}(\boldsymbol{\Psi}(\check{\mathcal{S}}), \text{loss}(\hat{\mathcal{S}}, \check{\mathcal{S}}))$
  $(\hat{\mathbf{Y}}_{\text{new}}, \mathsf{V}_{\min}) \leftarrow \text{argmin}_{\hat{\mathbf{Y}}} \mathbb{E}_{\check{\mathbf{Y}} \sim \check{\mathsf{P}}}[\text{loss}(\hat{\mathbf{Y}}, \check{\mathbf{Y}})]$
  if $(\hat{\mathsf{V}} \neq \mathsf{V}_{\min})$ **then** $\hat{\mathcal{S}} \leftarrow \hat{\mathcal{S}} \cup \hat{\mathsf{Y}}_{\text{new}}$
until $\check{\mathsf{V}} = \mathsf{V}_{\max} = \hat{\mathsf{V}} = \mathsf{V}_{\min}$
return $(\check{\mathcal{S}}, \hat{\mathcal{S}}, \hat{\mathsf{P}}, \check{\mathsf{P}})$

the predictor's best response is a single strategy(a pair of permutation vectors) that gives the minimal expected loss against $\check{\mathsf{P}}$.The problem can be written as:

$$\underset{\hat{\pi}_1, \hat{\pi}_2}{\text{argmin}}\, \mathbb{E}_{\check{\mathsf{P}}} \left[\frac{1}{n} \sum_{i}^{n} 1_{\hat{\pi}_{1,i} \neq \check{\pi}_{1,i}}(\hat{\pi}_{1,i}, \check{\pi}_{1,i}) \vee 1_{\hat{\pi}_{2,i} \neq \check{\pi}_{2,i}}(\hat{\pi}_{2,i}, \check{\pi}_{2,i}) \right]. \tag{4.22}$$

To better solve Eq. (Equation 4.22), let us first define a tensor representation of permutation π_1 and π_2 as $\mathbf{Y}(\pi_1, \pi_2) \in \mathbb{R}^{n \times n \times n}$ (or simply $\mathbf{Y}$) where the value of its cell $Y_{i,j,k}$ is 1 when $\pi_{1,i} = j$ and

$\pi_{2,i} = k$, and 0 otherwise. If Y representing a perfect 3D matching, each plane in any direction of $\mathbf{Y}$ can only have one entry of 1. Then Eq. (Equation 4.22) can be written as

$$\begin{aligned}
&\min_{\hat{\mathbf{Y}}} \mathbb{E}_{\check{p}} \left[< \hat{\mathbf{Y}}, \check{\mathbf{Y}} > \right] \\
=&\min_{\hat{\mathbf{Y}}} \sum_{t=1}^{l} < \hat{\mathbf{Y}} \cdot \check{p}_t \check{\mathbf{Y}}_t > \\
\text{s.t. : } &\sum_{i,j} \hat{\mathbf{Y}}_{i,j,k} = 1, \forall k \in [n] \\
&\sum_{j,k} \hat{\mathbf{Y}}_{i,j,k} = 1, \forall i \in [n] \\
&\sum_{i,k} \hat{\mathbf{Y}}_{i,j,k} 1, \forall j \in [n] \\
&\hat{\mathbf{Y}}_{i,j,k} \in \{0, 1\}.
\end{aligned} \tag{4.23}$$

where $\langle \cdot, \cdot \rangle$ denotes the Frobenius inner product between two tensors, i.e., $\langle A, B \rangle = \sum_{i,j,k} A_{i,j,k} B_{i,j,k}$.So what we need to find is

$$\begin{aligned}
&\operatorname*{argmin}_{\hat{\mathbf{Y}}} < \hat{\mathbf{Y}}, \sum_{t=1}^{l} \check{p}_t \check{\mathbf{Y}}_t > \\
\text{s.t. : } &\sum_{i,j} \hat{\mathbf{Y}}_{i,j,k} = 1, \forall k \in [n] \\
&\sum_{j,k} \hat{\mathbf{Y}}_{i,j,k} = 1, \forall i \in [n] \\
&\sum_{i,k} \hat{\mathbf{Y}}_{i,j,k} 1, \forall j \in [n] \\
&\hat{\mathbf{Y}}_{i,j,k} \in \{0, 1\}.
\end{aligned} \tag{4.24}$$

This is a 0-1 linear programming problem which is NP-complete (63). But there are lot of approximation method to solve it.

4.2.3.2 Adversary's best responses:

Now let's look at the adversary's best response. Given the predictor's distribution over strategies, $\hat{P}$(assuming there are l strategies in it), the adversary's best response is a single strategy(a pair of permutation vectors) that gives the maximal expected loss under $\hat{P}$. Comparing with predictor's best response, it also depends on the Lagrangian potential terms:

$$\underset{\check{\pi}_1,\check{\pi}_2}{\operatorname{argmax}}\, \mathbb{E}_{\check{p}}\left[\frac{1}{n}\sum_{i=1}^{n} 1_{\hat{\pi}_{1,i}\neq\check{\pi}_{1,i}}(\hat{\pi}_{1,i},\check{\pi}_{1,i}) \vee 1_{\hat{\pi}_{2,i}\neq\check{\pi}_{2,i}}(\hat{\pi}_{2,i},\check{\pi}_{2,i}) + \theta\cdot\left(\phi_i(\check{\pi}_{1,i},\check{\Pi}_{2,i},\mathbf{x})\right)\right]. \tag{4.25}$$

Using the permutation tensor $\mathbf{Y}$ again, we can rewrite Eq. (Equation 4.25) as

$$\begin{aligned}
&\underset{\check{Y}}{\operatorname{argmax}} < \sum_{t=1}^{l} \hat{p}_t \hat{\mathbf{Y}}_t, \check{\mathbf{Y}} > + \Psi(\check{\mathbf{Y}}) \\
\text{s.t. : } &\sum_{i,j}\sum_{i,j} \check{\mathbf{Y}}_{i,j,k} = 1, \forall k \in [n] \\
&\sum_{j,k}\sum_{j,k} \check{\mathbf{Y}}_{i,j,k} = 1, \forall i \in [n] \\
&\sum_{i,k}\sum_{i,k} \check{\mathbf{Y}}_{i,j,k} = 1, \forall j \in [n] \\
&\check{\mathbf{Y}}_{i,j,k} \in \{0,1\}.
\end{aligned} \tag{4.26}$$

where $\Psi(\check{\mathbf{Y}})$ is the Lagrangian potential got from $\check{\mathbf{Y}}$. Suppose we select features such that the Lagrangian potential is additive from each triples in the 3 dimensional matching, it means that there exist constant tensor A (given θ and X)such that $\Psi(\check{\mathbf{Y}}) =< A \cdot \check{\mathbf{Y}} >$, then we have the problem:

$$\begin{aligned}
&\underset{\check{Y}}{\text{argmax}} < (A + \sum_{t=1}^{l} \hat{p}_t \hat{\mathbf{Y}}_t), \check{\mathbf{Y}} > \\
\text{s.t. : } &\sum_{i,j} \sum_{i,j} \check{\mathbf{Y}}_{i,j,k} = 1, \forall k \in [n] \\
&\sum_{j,k} \sum_{j,k} \check{\mathbf{Y}}_{i,j,k} = 1, \forall i \in [n] \\
&\sum_{i,k} \sum_{i,k} \check{\mathbf{Y}}_{i,j,k} = 1, \forall j \in [n] \\
&\check{\mathbf{Y}}_{i,j,k} \in \{0, 1\}.
\end{aligned} \tag{4.27}$$

Like predictor's best response, this is also a 0-1 linear programming problem and can be solved using approximation methods.

4.2.4 Parameter Learning

To learn the θ in this problem, we also use AdaGrad (56) and compute the gradients $\mathbf{g}$:

$$\sum_{i=1}^{n} \left(\phi_i(\check{\Pi}_{1,i}, \check{\Pi}_{2,i}, \mathbf{x}) - \phi_i(\Pi_{1,i}, \Pi_{2,i}, \mathbf{x}) \right) \tag{4.28}$$

After learning all the parameters throught the training process, the prediction part will be similar as that in multiclass Classification problem.

4.2.5 Fisher Consistency

It has the same issue as the multiclass classification problem because approximation methods are used in finding the best responses. So we need further studies about its Fisher consistency.

4.3 Conclusion

We can try to use the traditional double oracle method to solve the NP-hard problems, as long as there are efficient approximation methods to handle them and the NP hardness always lies in the procedure of find the best responses.In the double oracle, finding the best responses will be done quite a lot of times depending on the convergence rate of θ and the size of equilibrium in each round of solving the inner game. It may cause the biases caused by the approximation methods accumulate a intolerable degree and we currently have not found a theory to estimate whether it is Fisher consistent. Although the experiment results show that this method can outperform SVM-Struct, we think it worthy to explore theoretical guarantees.

CHAPTER 5

MARGINAL DISTRIBUTION METHOD

(This chapter contains contents in paper "Adversarial Learning for 3D Matching. (4)", In Conference on Uncertainty in Artificial Intelligence (pp. 869-878). PMLR. The copyright policy for author reusing of them can be found in Appendix A)

Instead of directly solving the intractable problems through find the best response. There are some problems in which we can apply a marginal distribution method on them. The key idea is that the loss of these problem may only rely on certain marginal features of the data instead of all the features. By converting the problem into the marginal space, the size of the problem can be dramatically decreased. Although you may need to handle some other negative consequences somewhere else.

5.1 Background

For the 3D matching problem, using the double oracle method leads to solving 3D matching problems multiple times. Following (5), we can directly optimize on marginal distribution to significantly improves the training efficiency, as all quantities that we are interested in only rely on the marginal probabilities of the permutations.

Similar as the permutation tensors we defined in section 4.2.3.1, we can do the same for each feature function $\phi_i^{(l)}(\pi_{1,i}, \pi_{2,i}, x)$ by denoting its matrix representation as $\mathbf{X}_o$ whose (i, j, k)-th cell represents the o-th entry of $\phi_i(j, k, x)$. Then, for a given distribution of permutations $P(\pi_1, \pi_2)$, we denote the marginal probabilities of matching $i \in A$ with $j \in B$ and $k \in C$ as $p_{i,j,k} \triangleq P(\pi_{1,i} = j, \pi_{2,i} = k)$.

We let $\mathbf{P} = \sum_{\pi} P(\pi_1, \pi_2)\mathbf{Y}(\pi_1, \pi_2)$ be the predictor's marginal probability tensor where its (i, j, k) cell represents $\hat{P}(\hat{\pi}_{1,i} = j, \hat{\pi}_{2,i} = k)$, and similarly let $\mathbf{Q}$ be the adversary's marginal probability tensor (based on $\check{P}$).

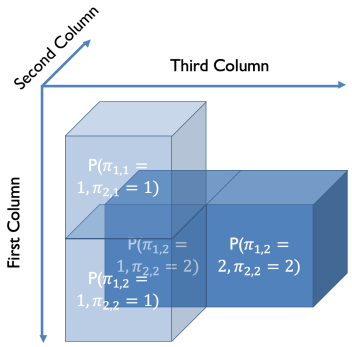

Figure 7: Marginal Tensor P

The size of this marginal matrices grows cubically ($\mathcal{O}(n^3)$), which is much smaller than the one of the original game matrix ($\mathcal{O}(n!^2)$).

By replacing $\hat{P}(\hat{\pi}_1, \hat{\pi}_2)$ and $\check{P}(\check{\pi}_1, \check{\pi}_2)$ with the matrix notation above, Eq. (Equation 4.18) can be rewrite as a minimax over marginal probability tensors $\mathbf{P}$ and $\mathbf{Q}$. The constraints are also reformed, and we have:

$$\begin{aligned}
&\min_{\theta} \mathbb{E}_{\mathbf{X},\mathbf{Y}\sim\tilde{P}} \min_{\mathbf{P}\geq\mathbf{0}} \max_{\mathbf{Q}\geq\mathbf{0}} \Big[1 - \frac{1}{n} \langle \mathbf{P}, \mathbf{Q} \rangle \\
&\qquad + \langle \mathbf{Q} - \mathbf{Y}, \textstyle\sum_o \theta_o \mathbf{X}_o \rangle \Big] \\
\text{s.t. : } &\sum_{i,j} \mathbf{P}_{i,j,k} = \sum_{i,j} \mathbf{Q}_{i,j,k} = 1, \forall k \in [n] \\
&\sum_{j,k} \mathbf{P}_{i,j,k} = \sum_{j,k} \mathbf{Q}_{i,j,k} = 1, \forall i \in [n] \\
&\sum_{i,k} \mathbf{P}_{i,j,k} = \sum_{i,k} \mathbf{Q}_{i,j,k} = 1, \forall j \in [n].
\end{aligned} \tag{5.1}$$

where $\langle \cdot, \cdot \rangle$ denotes the Frobenius inner product between two tensors, i.e., $\langle A, B \rangle = \sum_{i,j,k} A_{i,j,k} B_{i,j,k}$. We call the tensors $\mathbf{P}$ and $\mathbf{P}$ that satisfy the constrains in Eq. (Equation 5.1) as hyperplanar stochastic tensors.

To show Equation 5.1 is the same as Equation 4.17, firstly, let's look at the loss part. We have:

$$
\begin{aligned}
&\mathbb{E}_{\substack{\hat{\Pi}_1,\hat{\Pi}_2|\mathbf{x}\sim\hat{P}\\ \check{\Pi}_1,\check{\Pi}_2|\mathbf{x}\sim\check{P}}}\left[\text{loss}(\hat{\Pi}_1,\hat{\Pi}_2,\check{\Pi}_1,\check{\Pi}_2)\right] \\
=&\sum_{\hat{\pi}_1,\hat{\pi}_2}\sum_{\check{\pi}_1,\check{\pi}_2}\hat{P}(\hat{\pi}_1,\hat{\pi}_2)\check{P}(\check{\pi}_1,\check{\pi}_2)\text{loss}(\hat{\Pi}_1,\hat{\Pi}_2,\check{\Pi}_1,\check{\Pi}_2) \\
=&\frac{1}{n}\sum_{i}\sum_{\hat{\pi}_1,\hat{\pi}_2}\sum_{\check{\pi}_1,\check{\pi}_2}\hat{P}(\hat{\pi}_1,\hat{\pi}_2)\check{P}(\check{\pi}_1,\check{\pi}_2) \\
&1_{\hat{\pi}_{1,i}\neq\check{\pi}_{1,i}}(\hat{\pi}_{1,i},\check{\pi}_{1,i})\vee 1_{\hat{\pi}_{2,i}\neq\check{\pi}_{2,i}}(\hat{\pi}_{2,i},\check{\pi}_{2,i}) \\
&=\frac{1}{n}\sum_{i}\sum_{\hat{\pi}_1,\hat{\pi}_2}\sum_{\check{\pi}_1,\check{\pi}_2}\hat{P}(\hat{\pi}_1,\hat{\pi}_2)\check{P}(\check{\pi}_1,\check{\pi}_2) \\
&[1-1_{\hat{\pi}_{1,i}=\check{\pi}_{1,i}}(\hat{\pi}_{1,i},\check{\pi}_{1,i})\wedge 1_{\hat{\pi}_{2,i}=\check{\pi}_{2,i}}(\hat{\pi}_{2,i},\check{\pi}_{2,i})] \\
=&1-\frac{1}{n}\sum_{i}\sum_{\hat{\pi}_1,\hat{\pi}_2}\sum_{\check{\pi}_1,\check{\pi}_2}\hat{P}(\hat{\pi}_1,\hat{\pi}_2)\check{P}(\check{\pi}_1,\check{\pi}_2) \\
&1_{\hat{\pi}_{1,i}=\check{\pi}_{1,i}}(\hat{\pi}_{1,i},\check{\pi}_{1,i})\wedge 1_{\hat{\pi}_{2,i}=\check{\pi}_{2,i}}(\hat{\pi}_{2,i},\check{\pi}_{2,i}) \\
=&1-\frac{1}{n}\sum_{i}\sum_{j}\sum_{k}\hat{P}(\hat{\pi}_{1,i}=j,\hat{\pi}_{2,i}=k)\hat{P}(\check{\pi}_{1,i}=j,\check{\pi}_{2,i}=k) \\
=&1-\frac{1}{n}\sum_{i}\sum_{j}\sum_{k}\mathbf{P}_{i,j,k}\mathbf{Q}_{i,j,k} \\
=&1-\frac{1}{n}\langle\mathbf{P},\mathbf{Q}\rangle
\end{aligned}
$$

Then, for the potential part:

$$
\begin{aligned}
&\left\langle \mathbf{Q} - \mathbf{Y}, \sum_{o} \theta_o \mathbf{X}_o \right\rangle \\
=&\sum_{i} \sum_{j} \sum_{k} (\mathbf{Q}_{i,j,k} - \mathbf{Y}_{i,j,k}) \sum_{o} \theta_o \mathbf{X}_{o,i,j,k} \\
=&\sum_{i} \sum_{j} \sum_{k} \Big(\check{P}(\check{\pi}_{1,i} = j, \check{\pi}_{2,i} = k) \\
&- 1_{\pi_{1,i}=j \wedge \pi_{2,i}=k}(\pi_{1,i}, \pi_{2,i}) \Big) \cdot \theta^T \cdot \phi_i(\check{\pi}_{1,i}, \check{\pi}_{2,i}, x) \\
=&\sum_{i} \Bigg(\mathbb{E}_{\substack{\hat{\Pi}_1, \hat{\Pi}_2 | \mathbf{x} \sim \hat{P} \\ \check{\Pi}_1, \check{\Pi}_2 | \mathbf{x} \sim \check{P}}} \Big[\theta^T \cdot \phi_i(\check{\pi}_{1,i}, \check{\pi}_{2,i}, x) \Big] \\
&- \theta^T \cdot \phi_i(\pi_{1,i}, \pi_{2,i}, x) \Bigg) \\
=&\mathbb{E}_{\substack{\hat{\Pi}_1, \hat{\Pi}_2 | \mathbf{x} \sim \hat{P} \\ \check{\Pi}_1, \check{\Pi}_2 | \mathbf{x} \sim \check{P}}} \Big[\theta^T \cdot \sum_{i} \Big(\phi_i(\check{\pi}_{1,i}, \check{\pi}_{2,i}, x) - \phi_i(\pi_{1,i}, \pi_{2,i}, x) \Big) \Big]
\end{aligned}
$$

5.1.1 Optimization

To solve the marginal version of the problem, we try to adjust the order of the parameters we need to learn. By strong duality, we can pick **Q** as most external variable and push it to the left most part of the objective function. To smooth the objective, we add a strongly convex proxy-function to both **P** and **Q** as well as a regularization penalty to the parameter θ to prevent overfitting in our model. Also the

empirical expectation in Eq. (Equation 5.1) can be replaced by the average over training samples. Then we have the following optimization:

$$\begin{aligned}
\max_{\mathbf{Q}\geq\mathbf{0}} \min_{\theta} \frac{1}{m} \sum_{l=1}^{m} \min_{\mathbf{P}^{(l)}\geq\mathbf{0}} \Big[& \Big\langle \mathbf{Q}^{(l)} \\
& -\mathbf{Y}^{(l)}, \textstyle\sum_o \theta_o \mathbf{X}_o^{(l)} \Big\rangle - \frac{1}{n} \Big\langle \mathbf{P}^{(l)}, \mathbf{Q}^{(l)} \Big\rangle \\
& + \tfrac{\mu}{2} \|\mathbf{P}^{(l)}\|_F^2 - \tfrac{\mu}{2} \|\mathbf{Q}^{(l)}\|_F^2 \Big] + \tfrac{\lambda}{2} \|\theta\|_2^2 \\
\text{s.t. : } & \sum_{i,j} \mathbf{P}_{i,j,k}^{(l)} = \sum_{i,j} \mathbf{Q}_{i,j,k}^{(l)} = 1, \forall k \in [n] \\
& \sum_{j,k} \mathbf{P}_{i,j,k}^{(l)} = \sum_{j,k} \mathbf{Q}_{i,j,k}^{(l)} = 1, \forall i \in [n] \\
& \sum_{i,k} \mathbf{P}_{i,j,k}^{(l)} = \sum_{i,k} \mathbf{Q}_{i,j,k}^{(l)} = 1, \forall j \in [n],
\end{aligned} \tag{5.2}$$

where m is the number of 3D matching problems in the training set, λ is the regularization penalty parameter, μ is the smoothing penalty parameter, and $\|\mathcal{A}\|_F$ denotes the Frobenius norm of tensor $\mathcal{A}$. The superscript (l) in $\mathbf{P}^{(l)}, \mathbf{Q}^{(l)}, \mathbf{X}^{(l)}$, and $\mathbf{Y}^{(l)}$ refers to the l-th example in the training set.

In Eq. Equation 5.2, the inner minimization over θ and $\mathbf{P}$ can then be solved independently when $\mathbf{Q}$ is given. For θ we have a closed-form solution:

$$\theta_k^* = -\frac{1}{\lambda m} \sum_{l=1}^{m} \Big\langle \mathbf{Q}^{(l)} - \mathbf{Y}^{(l)}, \mathbf{X}_o^{(l)} \Big\rangle . \tag{5.3}$$

For $\mathbf{P}$, we can solve it independently for each training sample l:

$$\begin{aligned}
\mathbf{P}^{(l)*} &= \underset{\{\mathbf{P}^{(l)} \geq \mathbf{0}\}}{\operatorname{argmin}} \tfrac{\mu}{2}\|\mathbf{P}^{(l)}\|_F^2 - \frac{1}{n}\left\langle \mathbf{P}^{(l)}, \mathbf{Q}^{(l)} \right\rangle \\
&= \underset{\{\mathbf{P}^{(l)} \geq \mathbf{0}\}}{\operatorname{argmin}} \|\mathbf{P}^{(l)} - \tfrac{1}{n\mu}\mathbf{Q}^{(l)}\|_F^2 \\
\text{s.t. : } & \sum_{i,j} \mathbf{P}^{(l)}_{i,j,k} = 1, \forall k \in [n]; \sum_{j,k} \mathbf{P}^{(l)}_{i,j,k} = 1, \forall i \in [n] \\
& \sum_{i,k} \mathbf{P}^{(l)}_{i,j,k} = 1, \forall j \in [n].
\end{aligned} \tag{5.4}$$

This minimization is equivalent to projecting the tensor $\frac{1}{n\mu}\mathbf{Q}^{(l)}$ to the set of hyperplanar stochastic tensors. We solve this projection in the next section.

Now only $\mathbf{Q}$ is left. Given the solution of the inner optimization problems, we can then use the Quasi-Newton algorithm (64) to find the best $\mathbf{Q}$. After we achieve the adversary's optimal marginal probability $\mathbf{Q}^*$, we can use Eq. (Equation 5.3) to get θ^*, which is used in the prediction step.

5.1.2 Alternating Direction Method Of Multipliers

Before we go to the next step, let's first understand the alternating direction method of multipliers (ADMM) technique (65; 66) is a powerful tool for a problem that can be divided into a summation of two functions that only rely on two independent sets of parameters. The essential idea of the ADMM method is that for optimization problems with linear constraints and convex objective function, if the

objective function and constraints are both linearly separable when we divide the target variables into subgroups:

$$\min_{x,z} f(x) + g(z) \text{ s.t. : } Ax + Bz = c,$$

then the original problem can then be solved by the following step by step updating approach:

$$\begin{aligned} x^{k+1} &= \operatorname*{argmin}_{x} L_\rho(x, z^k, y^k) \\ z^{k+1} &= \operatorname*{argmin}_{z} L_\rho(x^{k+1}, z, y^k) \\ y^{k+1} &= y^k + \rho(Ax^{k+1} + Bz^{k+1} - c), \end{aligned}$$

where L_ρ is the Lagrangian of original problem plus a L_2 norm $\frac{\rho}{2}||Ax + Bz - c||_2^2$ with Lagrangian parameter y with the preset ADMM penalty parameter ρ. It can also expand to conditions in which variables are divided into more than two groups (67).

5.1.3 Hyperplanar Stochastic Tensors Projection

The hyperplanar stochastic tensors constraint can be solved using ADMM by dividing it into three sets of constraints $C_1 : \sum_{i,j} P_{i,j,k} = 1, \forall k \in [n]$ and $\mathbf{P} \geq \mathbf{0}$, $C_2 : \sum_{j,k} P_{i,j,k} = 1, \forall i \in [n]$ and $\mathbf{P} \geq \mathbf{0}$

and $C_3 : \sum_{i,k} P_{i,j,k} = 1, \forall j \in [n]$ and $\mathbf{P} \geq \mathbf{0}$. By adding two additional auxiliary variables S and T, Eq. Equation 5.4 can be rewrite as:

$$\begin{aligned} \min_{\mathbf{P},\mathbf{S},\mathbf{T}} \ & \tfrac{1}{2}\|\mathbf{P}-\mathbf{R}\|_F^2 + \tfrac{1}{2}\|\mathbf{S}-\mathbf{R}\|_F^2 + \tfrac{1}{2}\|\mathbf{T}-\mathbf{R}\|_F^2 + \\ & I_{C_1}(\mathbf{P}) + I_{C_2}(\mathbf{S}) + I_{C_3}(\mathbf{T}) \\ & \text{s.t. : } \mathbf{P}-\mathbf{S} = 0 \text{ and } \mathbf{P}-\mathbf{T} = 0, \end{aligned} \tag{5.5}$$

where $I_C(x)$ is an indicator function whose value is 0 when x satisfy logical expression C, otherwise it is inf. The augmented Lagrangian for this optimization is:

$$\begin{aligned} \mathcal{L}_\rho(\mathbf{P},\mathbf{S},\mathbf{T},\mathbf{W_1},\mathbf{W_2}) &= \tfrac{1}{2}\|\mathbf{P}-\mathbf{R}\|_F^2 + \tfrac{1}{2}\|\mathbf{S}-\mathbf{R}\|_F^2 \\ &+ \tfrac{1}{2}\|\mathbf{T}-\mathbf{R}\|_F^2 + I_{C_1}(\mathbf{P}) + I_{C_2}(\mathbf{S}) \\ &+ I_{C_3}(\mathbf{T}) + \tfrac{\rho}{2}\|\mathbf{P}-\mathbf{S}+\mathbf{W_1}\|_F^2 \\ &+ \tfrac{\rho}{2}\|\mathbf{P}-\mathbf{T}+\mathbf{W_2}\|_F^2 \end{aligned} \tag{5.6}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are two scaled dual variable. Since all the tensor operations here are just element wise operation, we can actually vectorise all the tensors without changing the results. Suppose all the variables in **P**,**S**,**R** are aligned in the order of $\mathbf{P}_{1,1,1}, \mathbf{P}_{1,1,2}, \cdots, \mathbf{P}_{1,1,n}, \mathbf{P}_{1,2,1}, \cdots, \mathbf{P}_{1,2,n}, \cdots, \mathbf{P}_{n,n,n}$

continued by elements in $\mathbf{S}$ and $\mathbf{T}$, and let us call this variable x. Then the constraints $\mathbf{P} - \mathbf{S} = 0$ and $\mathbf{P} - \mathbf{T} = 0$ can be combined into the form $\mathbf{A}x = 0$, where $\mathbf{A}$ looks like:

$$A = \begin{bmatrix} 1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & -1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & -1 \end{bmatrix}$$

This constraint can lead to a dual vector $\mathbf{w}$, then $\mathbf{W}_1$ and $\mathbf{W}_2$ are tensor version of the first and second half of $\mathbf{w}$ that generate from $\mathbf{P} - \mathbf{S} = 0$ and $\mathbf{P} - \mathbf{R} = 0$.

From the above formula, we can compute the update for $\mathbf{P}$ as:

$$
\begin{aligned}
\mathbf{P}^{t+1} &= \underset{\mathbf{P}}{\operatorname{argmin}}\, \mathcal{L}_\rho(\mathbf{P}, \mathbf{S}^t, \mathbf{T}^t, \mathbf{W}_1^t, \mathbf{W}_2^t) \\
&= \underset{\{\mathbf{P}|\mathbf{P}\in C_1\}}{\operatorname{argmin}}\, \tfrac{1}{2}\|\mathbf{P}-\mathbf{R}\|_F^2 + \tfrac{\rho}{2}\|\mathbf{P}-\mathbf{S}^t+\mathbf{W}_1^t\|_F^2 \\
&\quad + \tfrac{\rho}{2}\|\mathbf{P}-\mathbf{T}^t+\mathbf{W}_2^t\|_F^2 \\
&= \underset{\{\mathbf{P}|\mathbf{P}\in C_1\}}{\operatorname{argmin}}\, \tfrac{1}{2}\Big(\|\mathbf{P}\|_F^2 - 2\langle \mathbf{P}, \mathbf{R}\rangle\Big) \\
&\quad + \tfrac{\rho}{2}\Big(\|\mathbf{P}\|_F^2 - 2\langle \mathbf{P}, \mathbf{S}^t - \mathbf{W}_1^t\rangle\Big) \\
&\quad + \tfrac{\rho}{2}\Big(\|\mathbf{P}\|_F^2 - 2\langle \mathbf{P}, \mathbf{T}^t - \mathbf{W}_2^t\rangle\Big) \\
&= \underset{\{\mathbf{P}|\mathbf{P}\in C_1\}}{\operatorname{argmin}}\, \tfrac{1+2\rho}{2}\|\mathbf{P}\|_F^2 - \langle \mathbf{P}, \mathbf{R}\rangle \\
&\quad - \big\langle \mathbf{P}, \rho(\mathbf{S}^t - \mathbf{W}_1^t)\big\rangle - \big\langle \mathbf{P}, \rho(\mathbf{T}^t - \mathbf{W}_2^t)\big\rangle \\
&= \underset{\{\mathbf{P}|\mathbf{P}\in C_1\}}{\operatorname{argmin}}\, \|\mathbf{P}\|_F^2 - \tfrac{2}{1+2\rho}\big\langle \mathbf{P}, \mathbf{R} + \rho(\mathbf{S}^t + \mathbf{T}^t - \mathbf{W}_1^t - \mathbf{W}_2^t)\big\rangle \\
&= \underset{\{\mathbf{P}|\mathbf{P}\in C_1\}}{\operatorname{argmin}}\, \|\mathbf{P} - \tfrac{1}{1+2\rho}\big(\mathbf{R} + \rho\big(\mathbf{S}^t + \mathbf{T}^t \\
&\quad - \mathbf{W_1}^t - \mathbf{W_2}^t\big)\big)\|_F^2
\end{aligned}
$$

Key point for getting this is that modifying the object function by adding a constant or multiply a positive constant does not impact the process of finding the argmin.

The final form is a projection of $\frac{1}{1+2\rho}\left(\mathbf{R} + \rho\left(\mathbf{S}^t + \mathbf{T}^t - \mathbf{W_1}^t - \mathbf{W_2}^t\right)\right)$ to the tensor space that satisfy C_1. Since C_1 requires that all the elements in each slice in the direction of k sum up to 1, and each slice are independent from each other, the projection can also be done independently for each slice.

All the ADMM updates for other tensor variables can also be view as a projection, but from another direction. This technique has been studied previously, e.g., by (68).

So for $\mathbf{S}$ we have:

$$
\begin{aligned}
\mathbf{S}^{t+1} &= \underset{\mathbf{S}}{\operatorname{argmin}}\, \mathcal{L}_\rho(\mathbf{P}^{t+1}, \mathbf{S}, \mathbf{T}^t, \mathbf{W}_1^t, \mathbf{W}_2^t) \\
&= \underset{\{\mathbf{S}|\mathbf{S}\in C_2\}}{\operatorname{argmin}}\, \tfrac{1}{2}\|\mathbf{S}-\mathbf{R}\|_F^2 + \tfrac{\rho}{2}\|\mathbf{P}^{t+1}-\mathbf{S}+\mathbf{W}_1^t\|_F^2 \\
&= \underset{\{\mathbf{S}|\mathbf{S}\in C_2\}}{\operatorname{argmin}}\, \tfrac{1}{2}\Big(\|\mathbf{S}\|_F^2 - 2\langle \mathbf{S}, \mathbf{R}\rangle\Big) \\
&\quad + \tfrac{\rho}{2}\Big(\|\mathbf{S}\|_F^2 - 2\Big\langle \mathbf{S}, \mathbf{P}^{t+1}+\mathbf{W}_1^t\Big\rangle\Big) \\
&= \underset{\{\mathbf{S}|\mathbf{S}\in C_2\}}{\operatorname{argmin}}\, \tfrac{1+\rho}{2}\|\mathbf{S}\|_F^2 - \langle \mathbf{S}, \mathbf{R}\rangle \\
&\quad - \Big\langle \mathbf{S}, \rho(\mathbf{P}^{t+1}+\mathbf{W}_1^t)\Big\rangle \\
&= \underset{\{\mathbf{S}|\mathbf{S}\in C_2\}}{\operatorname{argmin}}\, \|\mathbf{S}\|_F^2 - \tfrac{1}{1+\rho}\Big\langle \mathbf{S}, \mathbf{R}+\rho(\mathbf{P}^{t+1}+\mathbf{W}_1^t)\Big\rangle \\
&= \underset{\{\mathbf{S}|\mathbf{S}\in C_2\}}{\operatorname{argmin}}\, \|\mathbf{S} - \tfrac{1}{1+\rho}\Big(\mathbf{R}+\rho\Big(\mathbf{P}^{t+1}+\mathbf{W_1}^t\Big)\Big)\|_F^2
\end{aligned}
$$

For $\mathbf{T}$ we have:

$$
\begin{aligned}
\mathbf{T}^{t+1} &= \underset{\mathbf{T}}{\operatorname{argmin}}\, \mathcal{L}_\rho(\mathbf{P}^{t+1}, \mathbf{S}^{t+1}, \mathbf{T}, \mathbf{W}_1^t, \mathbf{W}_2^t) \\
&= \underset{\{\mathbf{T}|\mathbf{T}\in C_3\}}{\operatorname{argmin}}\, \tfrac{1}{2}\|\mathbf{T}-\mathbf{R}\|_F^2 + \tfrac{\rho}{2}\|\mathbf{P}^{t+1}-\mathbf{T}+\mathbf{W}_2^t\|_F^2 \\
&= \underset{\{\mathbf{T}|\mathbf{T}\in C_3\}}{\operatorname{argmin}}\, \tfrac{1}{2}\Big(\|\mathbf{T}\|_F^2 - 2\langle \mathbf{T}, \mathbf{R}\rangle\Big) \\
&\quad + \tfrac{\rho}{2}\Big(\|\mathbf{T}\|_F^2 - 2\Big\langle \mathbf{T}, \mathbf{P}^{t+1}+\mathbf{W}_2^t\Big\rangle\Big) \\
&= \underset{\{\mathbf{T}|\mathbf{T}\in C_3\}}{\operatorname{argmin}}\, \tfrac{1+\rho}{2}\|\mathbf{T}\|_F^2 - \langle \mathbf{T}, \mathbf{R}\rangle \\
&\quad - \Big\langle \mathbf{T}, \rho(\mathbf{P}^{t+1}+\mathbf{W}_2^t)\Big\rangle \\
&= \underset{\{\mathbf{T}|\mathbf{T}\in C_3\}}{\operatorname{argmin}}\, \|\mathbf{T}\|_F^2 - \tfrac{1}{1+\rho}\Big\langle \mathbf{T}, \mathbf{R}+\rho(\mathbf{P}^{t+1}+\mathbf{W}_2^t)\Big\rangle \\
&= \underset{\{\mathbf{T}|\mathbf{T}\in C_3\}}{\operatorname{argmin}}\, \|\mathbf{T} - \tfrac{1}{1+\rho}\Big(\mathbf{R}+\rho\Big(\mathbf{P}^{t+1}+\mathbf{W_2}^t\Big)\Big)\|_F^2
\end{aligned}
$$

For $\mathbf{W}_1$ and $\mathbf{W}_2$ we can simply update them using the gradient.

We list all the update strategy here:

$$
\begin{aligned}
\mathbf{P}^{t+1} &= \mathrm{Proj}_{C_1}\Big(\tfrac{1}{1+2\rho}\left(\mathbf{R}+\rho\left(\mathbf{S}^t+\mathbf{T}^t-\mathbf{W}_1^t-\mathbf{W}_2^t\right)\right)\Big) \\
\mathbf{S}^{t+1} &= \mathrm{Proj}_{C_2}\Big(\tfrac{1}{1+\rho}\Big(\mathbf{R}+\rho\Big(\mathbf{P}^{t+1}+\mathbf{W}_1^t\Big)\Big)\Big) \qquad (5.7) \\
\mathbf{T}^{t+1} &= \mathrm{Proj}_{C_3}\Big(\tfrac{1}{1+\rho}\Big(\mathbf{R}+\rho\Big(\mathbf{P}^{t+1}+\mathbf{W}_2^t\Big)\Big)\Big) \qquad (5.8) \\
\mathbf{W}_1^{t+1} &= \mathbf{W}_1^t + \mathbf{P}^{t+1} - \mathbf{S}^{t+1} \qquad (5.9) \\
\mathbf{W}_2^{t+1} &= \mathbf{W}_2^t + \mathbf{P}^{t+1} - \mathbf{T}^{t+1}. \qquad (5.10)
\end{aligned}
$$

These updating steps are repeated until the primal and dual residual optimality is reached (66).

5.1.4 Prediction: Recovery From Marginal Distribution To 3D Matching

In the prediction step, we first use the θ^* we learned and the testing data to solve the inner optimization problem in Eq. Equation 5.4, giving us the predictor's best marginal distribution. After we get the marginal distribution, which may or may not correspond to an exact combination of 3D matchings, we still need to produce a 3D matching as the final prediction.

If the problem was bipartite matching, there is the Birkhoff–von Neumann theorem (69; 70) states that the convex hull of the set of $n \times n$ permutation matrices forms a convex polytope in $\mathbb{R}^{n^2}$ (known as the Birkhoff polytope B_n) in which points are doubly stochastic matrices, i.e., the $n \times n$ matrices with non-negative elements where each row and column must sum to one.

For bipartite matching, the Birkhoff–von Neumann theorem (69; 70) states that the convex hull of the set of $n \times n$ permutation matrices forms a convex polytope in $\mathbb{R}^{n^2}$ (known as the Birkhoff polytope B_n) in which points are doubly stochastic matrices, i.e., the $n \times n$ matrices with non-negative elements where each row and column must sum to one. This means that for bipartite matching, there is always a linear combination of matching tensors Y that sum to the marginal distribution.

This theorem tells us for bipartite matching, we can always find a linear combination of matching tensor Y that sum to the marginal distribution. And actually the prediction is done by finding the bipartite matching that maximizes the potential value, i.e., $\text{argmax}_{\mathbf{Y}} \langle \mathbf{Y}, \sum_k \theta_k \mathbf{X}_k \rangle$ which can be solved using the Hungarian algorithm.

However, this result does not generalize to 3D matching. (71) shows some extreme points of the multistochastic tensor are not convex combinations of permutation tensors. In their setting, each linear

grouped elements such as $\sum_i \mathbf{P}_{i,j,k} = 1 \, \forall j, k$. Though this tensor differs from ours, this implies that our marginal formulation is a relaxation from the original mixed strategy of permutations.

Never the less, we can try to find a $\mathbf{Y}$ that is close to the $\mathbf{P}$ we got and pursue the following problem:

$$\begin{aligned} &\underset{\mathbf{Y}}{\operatorname{argmin}} \ \|\mathbf{P} - \mathbf{Y}\|_F^2 \\ \text{s.t. : } &\sum_{i,j} \mathbf{Y}_{i,j,k} = 1, \forall k \in [n]; \sum_{j,k} \mathbf{Y}_{i,j,k} = 1, \forall i \in [n]; \\ &\sum_{i,k} \mathbf{Y}_{i,j,k} = 1, \forall j \in [n]; \mathbf{Y}_{i,j,k} \in \{0, 1\} \end{aligned} \tag{5.11}$$

$\mathbf{Y}$ is prediction we want and the whole problem is a integer quadratic programming problem. This problem is NP-hard (72), but there are approximation algorithms that can often be used for solving it in practice.

5.1.5 Fisher Consistency

In this problem the approximation part is no longer put in to the optimization process, but instead in recovery from the predicted marginal distribution to the expected combination of mixture strategies. Although we cannot use the existed theorem to prove the final predicted mixture strategies are Fisher consistency, comparing with SSVM that also try to optimize over the marginal distribution, we also have the advantage that we can guarantee the marginal distribution is Fisher consistent.

Notice that the the Hamming loss satisfy the condition that $\text{loss}(y, y) < \text{loss}(y', y)$ for all $y' \neq y$. And for the potential difference δ which act as f in theorem 3, it is additively defined as $\delta_{\pi_1,\pi_2} = \sum_{i=1}^{n} \delta_{\pi_1,\pi_2,i} = \sum_{i=1}^{n} \theta \cdot (\phi_i(\pi_{i,1}, \pi_{i,2}, x) - \phi_i(\pi'_{i,1}, \pi'_{i,2}, x))$. here π' is the grand truth. If we let $\delta_{\pi_1,\pi_2,i} = -1_{\pi_{1,i} \neq \pi^\diamond_{1,i}}(\pi_{1,i}, \pi^\diamond_{1,i}) \vee 1_{\pi_{2,i} \neq \pi^\diamond_{2,i}}(\pi_{2,i}, \pi^\diamond_{2,i})$ Where $\pi^\diamond$ is one of the optimal label. Then

$f^* + C(:, (\pi_1^\diamond, \pi_2^\diamond))$ is a uniform vector. Which leads to $\text{argmax}_{\pi_1,\pi_2} f^*(\pi_1, \pi_2, x) = \text{argmin}_{\pi_1,\pi_2}(C(:, (\pi_1^\diamond, \pi_2^\diamond))_{\pi_1,\pi_2} = (\pi_1^\diamond, \pi_2^\diamond)$

5.2 Conclusion

In this chapter, we explore the method of solving a adversarial learning problem in marginal distribution form on intractable problems. We can observe that this method can turn the training step into a very easy to solve problem but leave a hard problem at the prediction step. Considering that the training part is the most time consuming part in most machine learning tasks, this approach can be very help as long as a problem can be rewritten in marginal distribution form.

CHAPTER 6

EXPERIMENT

(This chapter contains contents in paper "Arc: Adversarial robust cuts for semi-supervised and multi-label classification. (3)", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 1905-1907). It also contains contents in paper "Adversarial Learning for 3D Matching. (4)", In Conference on Uncertainty in Artificial Intelligence (pp. 869-878). PMLR. The copyright policy for author reusing of them can be found in Appendix A)

6.1 Multiclass Classification with Pairwise Features

For the problem of multiclass classification with pairwise features, we introduce pairwise features on normal multiclass classification data set. When pairwise features are put in, the input samples are no longer viewed as independent from each other, but the whole input can be viewed as a single graph structured sample. The learning task in our model is to learn a function to map the unary and binary features to edge weights in a extended graph, so that after the matching, a multiway cut algorithm can best distinguish the nodes in the graph to the correct classes,i.e., making the correct nodes keep connecting to the virtual nodes that represent a particular class.

Using three datasets from the UCI repository (73; 74; 75). The characteristics of these datasets are summarized in Table III. Following (3), x_i will be directly used as unary features. We build a n times length(x) long vector and the ith section of the vector equals to x only if in $\phi(y, x)$, the valeu of y equals to the ith label. For pairwise features, we negate the value of the absolute difference of

TABLE III: Data Set for our multiclass classification with pairwise features.

Dataset	#Classes	#Features	#Training	#Testing
Vertebra	3	6	248	62
Vehicle	4	18	470	376
Libras	15	90	300	60
Ecoli	8	7	269	67
Glass	6	9	171	43
Breast Tissue	6	9	85	21
Image Segment	7	19	210	2100
Pen Digits	10	16	7494	3498
Satellite Image	6	36	4435	2000

the two corresponding nodes features $\boldsymbol{\sigma}_{i,j}(k, x) = \boldsymbol{\sigma}(x_i, x_j) = |x_i - x_j|$.In the common sense, if two nodes have close features, they turn to belong to the same class. This pairwise features allow adversarial assigning the same label to the two far away nodes, fulfilling our desire to make the adversarial increase the loss as possible as it can. We share the same unary and pairwise parameters across all edges so that these potentials can be applied to previously unseen samples at test time. During training time, we only incorporate labeled training samples. At testing time, we incorporate both the training set and the unlabeled testing set on which predictions are desired.

Since in this experiment we put additional features to normal classificiation problem, we fisrtly compare our method adversarial multiway cut (ADV-MC) with to state-of-art mulitclass classification methods without using pairwise features, they are k-nearest neighbor (k-NN) (76) and support vector machine with error-correcting output codes (SVM-ECOC) (77). For the data with pairwise feature, we apply the SVM-Struct(18; 13). For the support vector machine (SVM) methods, we used polynomial kernel.

Results are shown in Table IV.

TABLE IV: Average Accuracy of each algorithm.

Dataset	k-NN	SVM-ECOC	SVM-Struct	ADV
Vertebra	0.8226	0.8387	0.8567	0.8455
Vehicle	0.7154	0.7713	0.8133	0.8302
Libras	0.7167	0.8000	0.8045	0.8134
Ecoli	0.8690	0.8125	0.8872	0.8881
Glass	0.7336	0.6449	0.7452	0.7631
Breast Tissue	0.7453	0.5094	0.7315	0.7670
Image Segment	0.8905	0.9143	0.9226	0.9256
Pen Digits	0.9726	0.9551	0.9787	0.9804
Satellite Image	0.9000	0.8500	0.9235	0.9274

From the table we can find that by introducing pairwise feature, we indeed explored more information from the data and got better prediction results. However, this also means much more computational burden as we are forming complete graph for the whole data set. Finding a way to build a more efficient partial connected graph could be a promising future work. Our adversarial approach also get higher accuracy (1 - Hamming loss) for 8 of 9 cases than SVM-Struct.

6.2 3 Dimensional Matching

We apply our method on two different datasets. The first is a synthetic dataset that we can easily manipulate to test the property of the algorithm. Another one is the multiple object video tracking dataset (78).

Based on the assumption that there can be a linear combination of the feature that indicate the best matching, we create the synthetic data by uniformly generating raw data vectors with length l for all the $3n$ objects, and use them to construct the feature $\phi(\pi_1, \pi_2, x)$. To get the ground truth matching, we further uniformly generate a weight vector w that has the same size as $\phi(\pi_1, \pi_2, X)$. The permutation π_1, π_2 that leads to the highest value of $w^T \cdot \phi(\pi_1, \pi_2, X)$ will be used as the ground truth and it is

found through exhausted searching. For each object number n, we generate 10 groups of data sets with different w, and for each group, there are 50 triples of 3D matching samples.

For the video tracking task, we have a set of images (video frames) and a list of objects (bounding boxes) in each image alone with the ground truth matching between objects in frame t and objects in frame $t+1$. For the 3D matching task, we create a data sample by combining three consecutive frames in t, $t+1$ and $t+2$.

There are two groups of datasets: TUD datasets and ETH datasets with different numbers of objects and numbers of samples (frame triples). Table Table V contains the detailed information about the datasets. To make the training and testing samples more different, we pair up different datasets as training and testing sets.

The number of objects can be different in each frame, which is caused by objects entering and/or leaving. We address this by first expanding each frame to $3k$, where k is the maximal number of objects a frame can contain, to allow the cases that all the objects in each frame does not appear in the other frames. Then we add additional binary features indicating entering, leaving, occlusion in the middle or "staying invisible" (i.e., out of frame). For other pairwise and triple features that contain a virtual object, we assign similarity features as 0 and distance features as infinitely large to force real objects to turn to match with each other. The drawback of this setting is that it will ignore the cases such as two objects leave and one object enters. If the time interval between frames is small enough that at most one object leaves or enters a frame, this will not be a problem.

6.2.1 Feature Representation

The for the synthetic dataset, it is build in following way based on the assumption that there can be a linear combination of the feature that indicate the best matching.

1. Pick a size of the problem n, which is the number of objects that need to match. Pick a length of atom features l, it is the feature for each object in each stage.
2. Uniformly generate atom features X between 0 and 1. There will be $3m$ atom feature vectors need to be generated.
3. Using the atom features, we construct the feature $phi(\pi_1, \pi_2, x)$, details will be given later.
4. Uniformly generate a weight vector w that has the same size as $phi(\pi_1, \pi_2, X)$. The permutation π_1, π_2 that leads to the highest value of $w^T \cdot phi(\pi_1, \pi_2, X)$ will be used as the ground truth. Exhausted searching is used to find out the ground truth.

Let $X_{j,k}$ be the atom feature vector for the jth object in stage k. For the feature vector $phi(\pi_1, \pi_2, x)$, we define it additively as $phi(\pi_1, \pi_2, x) = \sum_{i=1}^{n} \phi_i(\pi_{1,i}, \pi_{2,i}, X)$. In $\phi_i(\pi_{1,i}, \pi_{2,i}, X)$ it contains the following parts

1. A copy of the atom features: $X_{i,1}$,$X_{\pi_{1,i},2}$ and $X_{\pi_{2,i},3}$.
2. Summation of each pair of atom features in the given tuple and the overall summation: $X_{i,1} + X_{\pi_{1,i},2}$, $X_{i,1} + X_{\pi_{2,i},3}$,$X_{\pi_{1,i},2} + X_{\pi_{2,i},3}$ and $X_{i,1} + X_{\pi_{1,i},2} + X_{\pi_{2,i},3}$.
3. The absolute difference between atom features: $|X_{i,1} - X_{\pi_{1,i},2}|$, $|X_{i,1} - X_{\pi_{2,i},3}|$ and $|X_{\pi_{1,i},2} - X_{\pi_{2,i},3}|$
4. Maximal and minimal L2 norm value of the atom features

5. Group differences $|X_{i,1} + X_{\pi_{1,i},2} - X_{\pi_{2,i},3}|$, $|X_{i,1} + X_{\pi_{2,i},3} - X_{\pi_{1,i},2}|$ and $|X_{\pi_{1,i},2} + X_{\pi_{2,i},3} - X_{i,1}|$

Since we have to exhaustively search for the ground truth, we only tried our method on small size data when $n = 4$ and $n = 5$. We generated 10 group of data set with different w, and for each group, there are 50 3 dimension matching samples in which 40 samples are used for training and 10 samples are used for testing.

To make comparison, we applied the SVM-Struct using the same feature. Notice that for SVM-Struct, we also only aim to learn an optimal marginal distribution. Directly solving the 3 dimensional matching problem will be NP-hard for both SVM-Struct and adversarial method.

For the video tracking problem, we use an existing feature representation (79) that uses six different types of features:

- Intersection over union (IoU) overlap ratio between bounding boxes. For pairs, IoU is area$(BB_i^{t_1} \cap BB_j^{t_2})$/ area$(BB_i^{t_1} \cup BB_j^{t_2})$, where BB_i^t denotes the bounding box of object i at time frame t. For triples, it is area$(BB_i^{t_1} \cap BB_j^{t_2} \cap BB_k^{t_3})$/ area$(BB_i^{t_1} \cup BB_j^{t_2} \cup BB_k^{t_3})$.
- Euclidean distance between object centers;
- 21 color histogram distance features (RGB) from the Bhattacharyaa distance, $\frac{1}{4}\ln\left(\frac{1}{4}\left(\frac{\sigma_p^2}{\sigma_q^2} + \frac{\sigma_q^2}{\sigma_p^2} + 2\right)\right) + \frac{1}{4}\left(\frac{(\mu_p - \mu_q)^2}{\mu_p^2 + \mu_q^2}\right)$, between distributions from the histograms of 7×3 blocks, in which p and q are two different distributions of the blocks at time frames t and $t+1$, μ and σ^2 are the mean and the variance of the distribution respectively; For triples we also use average value of the pairwise distance.

TABLE V: Video tracking dataset properties.

DATASET	# OBJECTS	# SAMPLES
TUD-CAMPUS	12	69
TUD-STADTMITTE	16	177
ETH-SUNNYDAY	18	352
ETH-BAHNHOF	34	998
ETH-PEDCROSS2	30	835

- 21 local binary pattern (LBP) (80) features from similar Bhattacharyaa distances and bounding box blocks; For triples we also use average value of the pairwise distance.
- Optical flow (motion) (81) between bounding boxes. For triples we also use average value of the pairwise distance of the affine transformation parameters.
- Four indicator variables (for *entering*, *leaving*, *hiding in the middle* and *staying invisible)*.

6.2.2 Experiment Setup

We apply two different adversarial approaches for 3D matching. One uses double oracle method(Adv3DDo) and the other one uses margin method (Adv3DMarg). To make the comparison with our method, we implement the SVM-Struct model (18; 19) based on (79) using `SVM-Struct` (82; 83) for both and two-stage marginal adversarial bipartite matching (2S-AdvMarg) proposed by (5). For the SVM-Struct, we also use it to predict the best marginal distribution. We use `minConf` (84) to perform the projected Quasi-Newton optimization. In the prediction part of SVM-Struct and Adv3DMarg, we used the Gurobi Mixed-Integer Programming solver. For 2S-AdvMarg, we simply apply it on frame t to $t+1$ and frame $t+1$ to $t+2$ separately, and pick out the matched triples. We use 5-fold cross validation to tune the regularization parameter (λ in adversarial matching, and C in SVM-Struct).

TABLE VI: The mean and standard deviation (in parenthesis) of the expected average accuracy for synthetic data.

# OBJECTS	2S-ADV MARG.	ADV3D MARG.	ADV3D DO	SVM-STRUCT
3	0.883 (0.06)	0.926 (0.05)	0.933 (0.06)	0.893 (0.05)
4	0.812 (0.07)	0.855 (0.09)	0.865 (0.08)	0.854 (0.06)
5	0.826 (0.09)	0.835 (0.09)	0.836 (0.08)	0.816 (0.08)
6	0.795 (0.08)	0.826 (0.09)	0.827 (0.09)	0.822 (0.07)

6.2.3 Results

Table Table VII and Table Table VIII provide the mean and the standard deviation of the average accuracy for synthetic and video tracking data. It is calculated by $1-\text{loss}_{\text{Hamming}}$. For the synthetic data we also try to use the marginal distribution directly to calculate the expected loss, loss $= \mathfrak{n}- < \hat{\mathsf{P}}, \mathsf{Y} >$, and the results are in Table VI. Comparing with Table VII, we can the the prediction we used is pretty close to the expected accuracy using mixed strategies. This synthetic data is design to have a linear indicator function for the ranking result, which makes the accuracy being very high.

We can see that three of the 3D matching algorithms are consistently better than the 2S-AdvMarg, indicating that directly solving the 3D matching problem can indeed improve the performance beyond simply applying the multistage bipartite matching. Unlike the case in bipartile matching in which the double oracle method can always out perform marginal method (5), double oracle method can achieve better performance on synthetic data with small number of samples and objects but failed in providing

TABLE VII: The mean and standard deviation (in parenthesis) of the average accuracy for synthetic data.

# OBJECTS	2S-ADV MARG.	ADV3D MARG.	ADV3D DO	SVM-STRUCT
3	0.852 (0.09)	0.885 (0.08)	0.891 (0.09)	0.893 (0.09)
4	0.812 (0.10)	0.855 (0.10)	0.858 (0.09)	0.833 (0.09)
5	0.815 (0.12)	0.827 (0.11)	0.827 (0.10)	0.802 (0.10)
6	0.779 (0.10)	0.808 (0.11)	0.810 (0.11)	0.800 (0.09)

higher accuracy when applied to large real data. The major causes for it could be that approximation methods were used during the finding the best response part in double oracle method and the error introduced by approximation was accumulated during multiple round of training. On the other hand, the marginal method only needs to do approximation at the prediction stage. Comparing with the results of multi-class classification in Table III in which adversarial double oracle can always outperform SSVM, it indicates that the approximation methods used in finding best response have a big impact on the performance of the double oracle based adversarial methods. To compare the accuracy with SVM-Struct, we use bold font to show the cases where Adv3DMarg outperformed with statistical significance. We can see that we have significantly better results on 4 pairs of the datasets and a bit better than SVM-Struct on the other ones. For the synthetic data, the accuracy decreases as the number of objects increases, which is reasonable as the problem becomes harder.

TABLE VIII: The mean and standard deviation (in parenthesis) of the average accuracy for video tracking.

TRAINING/ TESTING	2S-ADV MARG.	ADV3D MARG.	ADV3D DO	SVM-STRUCT
CAMPUS/ STADTMITTE	0.421 (0.07)	**0.453** (0.08)	0.431 (0.09)	0.424 (0.07)
STADTMITTE/ CAMPUS	0.452 (0.10)	0.478 (0.11)	0.465 (0.10)	0.470 (0.09)
BAHNHOF/ SUNNYDAY	0.552 (0.06)	**0.578** (0.05)	0.572 (0.06)	0.568 (0.05)
PEDCROSS2/ SUNNYDAY	0.535 (0.08)	**0.563** (0.08)	0.543 (0.09)	0.545 (0.10)
SUNNYDAY/ BAHNHOF	0.541 (0.15)	0.583 (0.17)	0.581 (0.17)	0.570 (0.18)
PEDCROSS2/ BAHNHOF	0.565 (0.10)	0.597 (0.13)	0.585 (0.12)	0.589 (0.16)
BAHNHOF/ PEDCROSS2	0.492 (0.11)	0.523 (0.11)	0.509 (0.10)	0.511 (0.13)
SUNNYDAY/ PEDCROSS2	0.499 (0.14)	**0.537** (0.12)	0.526 (0.13)	0.522 (0.14)

To further evaluate the robustness of the methods, we also tried to add a Gaussian white noise to the $w \cdot \phi$ part and generate polluted training data on the synthetic data. From the results in Figure Figure 8 that depict the case when $n = 5$, we can see that Adv3DMarg can slightly better keeping a high performance than SSVM.

To compare the running time, we list the time used on the video tracking dataset in Table Table IX. The predictions from Adv3DMarg and SSVM differ from that in 2S-AdvMarg, but since the prediction consumes much less time in our setting, we only focus on the training time. It shows that SSVM is faster, but Adv3DMarg is acceptable within this scale of data. And if we use double oracle, the speed is much

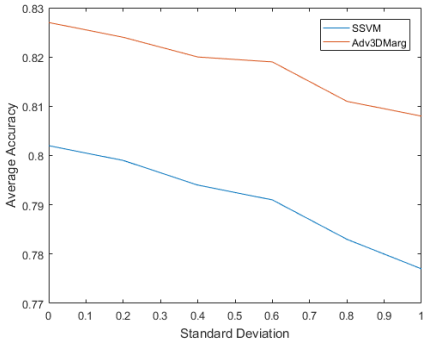

Figure 8: $n = 4$ Average accuracy when Gaussian noise added in $\{141, 233, 312, 424\}$

slower. The running time of Adv3DMarg grows roughly cubically in the number of objects, identical with the growth rate of the 3D tensor. This speed is much better than employing a CRF approach, which has running time that is high even for small problems. Unlike results for bipartite matching in which the SSVM tries to predict the matching directly and has an efficient direct method to solve the inner optimization method (5), SSVM also needs to solve the ADMM problem for 3D matchings. Thus, SSVM for 3D matching is much slower than the one in bipartite matching. However, it still has the speed benefit that may also caused by different tools for implementation, i.e., C++ for SSVM and MATLAB for our method.

6.3 Conclusion

We expand the adversarial approach for multi-binary label problem to the realm of multi-multiclass label problems. Through the analysis we succeed in forming the best response subproblem as a multi-cut problem. By using approximation methods to solve the NP-hard subproblem, we could efficiently train

TABLE IX: Running time (in seconds) with 50 samples

DATASET	# OBJECTS	ADV3DM-DO	ADV3DMARG	SSVM
CAMPUS	12	270.70	21.34	11.72
STADTMITTE	16	358.69	54.73	18.79
SUNNYDAY	18	399.86	73.22	22.69
PEDCROSS2	30	3037.91	357.70	146.12
BAHNHOF	34	2363.38	563.50	173.41

and predict. Although errors are brought in throw approximation, the adversarial multi-cut approach show good robust property and outperform standard SSVM and other competitors.

We also use adversarial learning to formulate the 3D matching learning problem. We tried both double oracle with direct approximation and marginal distribution formulation. Although the direct approximation does not reach very good results, the marginal distribution formulation did very well. Results of the average accuracy clearly show the improvement comparing with two stage bipartite matching approach. It also has better results over SSVM.It is a way that avoids directly solving the 3D matching problem and can efficiently train on both synthetic and real datasets.

We postpone the challenge of solving a NP-hard problem to the prediction stage. Although the practical results are promising, there is still no clear theoretical proof about the error bound by transforming the marginal tensor to an exact 3D matching. Building a more solid theoretical base for this method remains as important future works.

CHAPTER 7

CONCLUSION AND FUTURE WORKS

7.1 Conclustion

In this work, we expand the adversarial learning approach to two new field: multiclass classification with pairwise feature and 3 dimensional matching. They are different as one have a structured inputs and the other one has both structured inputs and outputs. On the other hand, for both of the learning tasks, there are a NP-hard sub-problem need to be solved.

To deal with these difficulties, we had two different routes. For solving multiclass classification with pairwise feature, we directly use approximate algorithm in the training process. As for solving 3 dimensional matching, we also tried to converge the problem into learning a marginal distribution. In this way, in the training process, we no longer need to face a hard problem, but left the hard problem in prediction. As training is the most time consuming part, it may save a lot of time.

We mainly compare our adversarial method with SVM-Struct. When dealing with easier problems where no approximation is needed. There are already many works showing both theoretically and practically, our method can outperform the original SVM-Struct in the sense of providing better prediction results, although may spend more time. Especially, we have establish a series of analytical tools for showing the adversarial approach is Fisher consistent. In this paper, for solving the learning problems with hard sub-problems, we also showed the adversarial approach may get better practical results.

7.2 Future Works

Never the less, the works in this paper are still incomplete and there are many future works need to do:

7.2.1 N Dimensional matching

We have expand the problem of bipartite matching to 3D matching. Actually we can try to used the same marginal trick to even higher dimension cases. When we do the n dimensional matching, suppose we still using the loss that is 1 as long as any of the element in one set does not meet the grand truth, then we can still model the problem using higher dimensional permutation Tensor and hyperplanar stochastic tensor. Since all the tensor operations in this model are actually all elementwise, we can still got formula Equation 5.1 and optimize it. And in the projection part we can use alternating direction method of multipliers (ADMM) that separate the object in n parts.

It may be worth to evaluate the exact computing time in these cases. But the biggest issue will be would the space of mixture of permutation tensor and the space of hyperplanar stochastic tensor the same? And is thee an efficient algorithm to cover a hyperplanar stochastic tensor to a mixture of permutation tensor? If these two questions have answers of yes, then it will prove P is equivalent to NP, so they will highly probably be false. But we still need a valid proof for it, or even figure out how much differences they have.

7.2.2 Numerical Bounds of Adversarial Learning Using Approximation Methods

There is still no theoretical guarantees for the performance of adversarial learning method when using approximation algorithm to solve the inner problems. If we can use marching approach like

what we did in solving the 3D matching problem, and avoid approximation in training stage, the biases introduced may not be a big issue. But if we are doing double oracle, the bias may be accumulated to an unacceptable level.

7.2.3 Adversarial Learning on Other Intractable Problems

In this thesis, there are only two intractable problems are shown and discussed. Although they gave us some insight and can represent certain type of intractable problems, the solution for them are highly specified and still cannot give us a overall evaluation about how adversarial learning on a variety of intractable problems. It can be worthy to apply it to other problems, and try to prove and summarize more common perperties of this method.

One option can be direct apply adversarial method on the video tracking problem. Preliminary attempts have been done on this direction. The biggest issue is that the loss function of video tracking requires a ranking of predicted trajectories and it is hard to directly apply this loss function into the adversarial learning model. If we define the Hamming loss here as

$$L_{Hamm}(\hat{Y}, \check{Y}) = \frac{1}{KM} \sum_{k=1}^{K} \sum_{m=1}^{M} \mathbb{I}_r(\hat{o}_{k,m}, \check{o}_{k,m}) \tag{7.1}$$

Then the loss of video tracking problem will be the Best Matched Hamming Loss

$$L_{B_Hamm}(\hat{Y}, \check{Y}) = min_{\pi} \frac{1}{KM} \sum_{k=1}^{K} \sum_{m=1}^{M} \mathbb{I}_A(\hat{o}_{k,m}, \check{o}_{\pi_k m}) \tag{7.2}$$

In the most simplified cases under the assumption that

1. Ground truth object bounding boxes are given and they can represent the object as o.

2. Each frame have the same number of objects and no new object in and no old object out.

3. The track sets have the same size and it is full(contents K tracks) and each trach has the same length and full(M objects).

we can define additive Markov feature/potential assumption:

$$\psi(Y, X) = \sum_{k=1}^{K} \psi(T_k, X) = \sum_{k=1}^{K} \left(\sum_{m=1}^{M} \psi_1(o_{k,m}, X) + \sum_{m=1}^{M-1} \psi_2(o_{k,m}, o_{k,m+1}, X) \right) \tag{7.3}$$

Note that when Y and each check in Y is full, this version of potential function does not care about which track a object is put in but only care about which adjacent objects are put in the same track together, so the unary feature will be meanless since each object feature will join the calculation once no matter what Y is, so the unary feature can be omitted

If we use Hamming loss with fixed matches for the prediction and ground truth tracks. We can get

7.2.3.1 Predictor's Best Response

Suppose there are I_1 candidate adversary's strategies:

$$\underset{\hat{Y}}{\operatorname{argmin}} \frac{1}{KM} \sum_{i_1=1}^{I_1} P_{i_1} \sum_{k=1}^{K} \sum_{m=1}^{M} \mathbb{I}_r(\hat{o}_{k,m}, \check{o}_{i_1,k,m}) \tag{7.4}$$

(7.5)

Since

$$\min_{\hat{Y}} \frac{1}{KM} \sum_{i_1=1}^{I_1} P_{i_1} \sum_{k=1}^{K} \sum_{m=1}^{M} \mathbb{I}_r(\hat{o}_{k,m}, \check{o}_{i,k,m}) \tag{7.6}$$

$$\Leftrightarrow \frac{1}{KM} \sum_{m=1}^{M} \min_{\{\hat{o}_{k,m}\}} \sum_{k=1}^{K} \sum_{i_1=1}^{I_1} \mathbb{I}_r(\hat{o}_{k,m}, \check{o}_{i_1,k,m}) \tag{7.7}$$

We can separately find out

$$\underset{\{\hat{o}_{k,m}\}}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{i_1=1}^{I_1} \mathbb{I}_r(\hat{o}_{k,m}, \check{o}_{i_1,k,m}) \tag{7.8}$$

Let δ_{i_2,i_3} means the i_3th object in the frame is put in i_2th track in $\hat{Y}$, then eq. (Equation 7.8) can be rewritten as

$$\underset{\{\delta_{i_2,i_3}\}}{\operatorname{argmin}} \sum_{i_2=1}^{K} \sum_{i_3=1}^{K} C_{i_2,i_3} \delta_{i_2,i_3} \tag{7.9}$$

$$\text{s.t.} \sum_{i_2=1}^{K} \delta_{i_2,i_3} = 1, \forall i_3, \tag{7.10}$$

$$\sum_{i_3=1}^{K} \delta_{i_2,i_3} = 1, \forall i_2. \tag{7.11}$$

$$\delta_{i_2,i_3} \in 0, 1, \forall i_2, i_3 \tag{7.12}$$

where $C_{i_2,i_3} = \sum_{i_1=1}^{I_1} P_{i_1} \mathbb{I}_r(b_{i_3,m}, \check{o}_{i_1,i_2,m})$

7.2.3.2 Adversarial's Best Response

Also suppose there are I_1 candidate predictor's strategies, we can write the adversarial's best response as:

$$\underset{\check{Y}}{\operatorname{argmax}} \frac{1}{KM} \sum_{i_1=1}^{I_1} P_{i_1} \sum_{k=1}^{K} \sum_{m=1}^{M} \mathbb{I}_r(\hat{o}_{i_1,k,m}, \check{o}_{k,m}) + \sum_{k=1}^{K} \left(\sum_{m=1}^{M} \psi_1(\check{o}_{k,m}, X) + \sum_{m=1}^{M} \psi_2(\check{o}_{k,m}, \check{o}_{k,m+1}, X) \right) \tag{7.13}$$

If we introduce $\delta_{i_2,i_3,m}$ as that in predictor's best response, and $\gamma_{i_4,i_5,m}$ as a variable indicating that $b_{i_4,m}$ is paired with $b_{i_5,m+1}$, it can be rewritten as:

$$\underset{\{\delta_{i_2,i_3,m}\},\{\gamma_{i_4,i_5,m}\}}{\operatorname{argmax}} \sum_{m=1}^{M}\sum_{i_2=1}^{K}\sum_{i_3=1}^{K} C_{i_2,i_3,m}\delta_{i_2,i_3,m} + \sum_{m=1}^{M-1}\sum_{i_4=1}^{K}\sum_{i_5=1}^{K} D_{i_4,i_5,m}\gamma_{i_4,i_5,m} \tag{7.14}$$

$$\text{s.t.} \sum_{i_2=1}^{K} \delta_{i_2,i_3,m} = 1,\ \forall i_3,\ m. \tag{7.15}$$

$$\sum_{i_3=1}^{K} \delta_{i_2,i_3,m} = 1,\ \forall\, i_2,\ m. \tag{7.16}$$

$$\sum_{i_4=1}^{K} \gamma_{i_4,i_5,m} = 1,\ \forall\, i_5,\ m. \tag{7.17}$$

$$\sum_{i_5=1}^{K} \gamma_{i_4,i_5,m} = 1,\ \forall\, i_4,\ m. \tag{7.18}$$

$$\sum_{i_2=1}^{K} \delta_{i_2,i_4,m}\,\delta_{i_2,i_5,m+1} = \gamma_{i_4,i_5,m},\ \forall\, i_4,\ i_5,\ m. \tag{7.19}$$

$$\delta_{i_2,i_3,m} \in 0,1,\ \forall\, i_2,\ i_3,\ m. \tag{7.20}$$

$$\gamma_{i_4,i_5,m} \in 0,1,\ \forall\, i_4,\ i_5,\ m. \tag{7.21}$$

where $C_{i_2,i_3,m} = \sum_{i_1=1}^{I_1} P_{i_1}\mathbb{I}_r(b_{i_3,m}, \eth_{i_1,i_2,m})$ and $D_{i_4,i_5,m} = \psi_2(b_{i_4,m}, b_{i_5,m+1}, X)$.

This is again an linear integral programming problem and can be solved using approximation method.

Solutions for general case or mapping from general cases that violate those assumptions are still need to be explored.

APPENDIX

A. COPYRIGHT

UAI/CoRR Copyright Permission

APPENDIX (Continued)

TITLE OF WORK: _____

AUTHOR(S): _____

DESCRIPTION OF MATERIAL: Paper in UAI 2017 proceedings

I hereby grant permission for AUAI and CoRR to include the above-named material (the *Material)* in the UAI and CoRR Digital Libraries.

☐ **Yes, I verify that I am the owner of copyright in this Material and have the authority to grant permission.**

I hereby release and discharge AUAI, CoRR, and other publication sponsors and organizers from any and all liability arising out of my inclusion in the publication, or in connection with the performance of any of the activities described in this document as permitted herein. This includes, but is not limited to, my right of privacy or publicity, copyright, patent rights, trade secret rights, moral rights or trademark rights.

All permissions and releases granted by me herein shall be effective in perpetuity unless otherwise stipulated, and extend and apply to the AUAI, CoRR, and its assigns, contractors, sublicensed distributors, successors and agents.

The following statement of copyright ownership will be displayed with the Material, unless otherwise specified: **"Copyright is held by the author/owner."**

In the event that any elements used in the Material contain the work of third-party individuals, I understand that it is my responsibility to secure any necessary permissions and/or licenses and will provide same in writing to AUAI and CoRR. If the copyright holder requires a citation to a copyrighted work, I have obtained the correct wording and have included it in the designated space in the text.

☐ No, I have not used third-party material.
☐ I have the necessary permission to use third-party material.

_____
*SIGNATURE (author/owner)

_____
PRINT NAME DATE

Signature block, use if more space is needed.

Note: Each contributor must submit a signed form, or indicate below that one agent signs for all co-authors. Rev. 3.11

☐ I am signing on behalf of all co-authors

APPENDIX (Continued)

Instructions for Authors

A lot of work needs to be done over a very short time between the arrival of the papers and the submission of the whole package. Please help us by reading and carefully following these instructions.

1. **Upload new paper/metadata:** To submit the camera-ready version of your paper, log into the confmaster web site `http://nips2008.confmaster.net`, click on "View Own Papers", click on the diskette with the red uparrow below it on the row that corresponds to your accepted paper, and upload your PostScript or PDF file. If the title or authors are different from those printed in this brochure, send an email to `nips@nec-labs.com` with the subject "Subject: NIPS paper *paperid* Title or Author Change", where *paperid* is the confmaster paper number.

   For document preparation instructions, see below. Remember to use the latest style files and to set them to generate a non-anonymous paper.

   All camera-ready papers and metadata must be uploaded to the website by 11:59PM Eastern Time, Friday, January 9, 2009.

2. **Author Agreement Form:** One author must complete and sign the author agreement form which appears on the last page of the NIPS program booklet (and is also available on the NIPS website). This year we are moving to electronic collection of these forms. Please scan the completed, signed form and convert it to PDF format. (Color is optional.) Email the PDF scan of the completed, signed form to `nips@nec-labs.com`. Use the subject "Subject: NIPS paper *paperid* Agreement Form", where *paperid* is the paper number assigned at papers.nips.cc. The signed, scanned author agreement form is due by Wednesday, January 7, 2009.

3. Make sure you have prepared your paper according to the NIPS style file. Use the official NIPS style files and resist temptations to redefine parameters such as `\textheight` and `\textwidth` to give yourself more space. There is a limit of 8 pages. Stay with the required font sizes. If you do not follow the formatting instructions we will ask you to resend your paper. This will only result in more work for yourself and for the publication chair. Links to the style files and detailled instructions are available on `http://ml.nec-labs.com/nips`.

4. Do not include in the front page of the paper any statement about "To appear in..." or about the NIPS category your paper belongs to. Do not force page numbers to appear.

5. Make sure your paper prints correctly in black and white. Do not rely on colors to convey information in figures. Avoid figures with dark grey curves on gray background.

6. Please use PostScript or PDF format with paper size "letter". Make sure that PDF files contain only Type-1 fonts using program `pdffonts` or using menu "File→Document Properties→Fonts" in Acrobat. Other fonts (like Type-3) might come from graphics files imported into the document. If you want to make sure your paper is printable, please use only Type-1 fonts.

   LaTeX users should try the following commands:

   ```
   dvips paper.dvi -o paper.ps -t letter -Ppdf -G0
   ps2pdf paper.ps
   ```

APPENDIX (Continued)

Contributing Authors' Letter of Agreement

(must be completed by one author per paper)

Title of article: _____

_____

Author(s): _____

_____

For the purpose of publication of the above article in the book tentatively entitled

Advances in Neural Information Processing Systems 21

I do warrant that we are the sole authors except for those portions shown to be in quotations;

I authorize the Neural Information Processing Systems Foundation to publish the article and to claim copyright in the book in which it is published for all editions, to license translations of the article as part of translations of the book as a whole, and also to license non exclusively reprints and electronic publications of the article;

I appoint the editor as our agent in everything pertaining to the publication of the article in this book;

I agree to read proof if requested to do so and otherwise to cooperate to the end of publication;

and I confirm that the article does not violate copyright or other proprietary right, that it contains no material that is libelous or in any way unlawful, that it has not previously been been published, in the whole or in part, except as I have advised the editor in writing, and that I will not now publish the article in any other edited volume without the consent of the Neural Information Processing Systems Foundation.

With this last exception, we retain the right to use the material in the article in any other writing of our own, whether in article or book form.

Signature: _____

Date: _____

Address: _____

_____

_____

AAAI Press Copyright Form

APPENDIX (Continued)

Title of Article/Paper:

Please type or print your name as you wish it to appear in print:

(Please read and sign Part A only, unless you are a government employee and created your article/paper as part of your employment. If your work was performed under Government contract, but you are not a Government employee, sign Part A and see item 5 under returned rights.)

PART A--Copyright Transfer Form

The undersigned, desiring to publish the above article/paper in a publication of the American Association for Artificial Intelligence, (AAAI), hereby transfer their copyrights in the above article/paper to the American Association for Artificial Intelligence (AAAI), in order to deal with future requests for reprints, translations, anthologies, reproductions, excerpts, and other publications.

This grant will include, without limitation, the entire copyright in the article/paper in all countries of the world, including all renewals and extensions thereof, and also the exclusive right to create and distributed electronic versions of the article/paper, to the extent that such right is not subsumed under copyright.

The undersigned warrants that they are the sole author and owner of the copyright in the above article/paper, except for those portions shown to be in quotations; that the article/paper is original throughout; and that the undersigned right to make the grants set forth above is complete and unencumbered.

If anyone brings any claim or action alleging facts that, if true, constitute a breach of any of the foregoing warranties, the undersigned will hold harmless and indemnify AAAI, their grantees, their licensees, and their distributors against any liability, whether under judgment, decree, or compromise, and any legal fees and expenses arising out of that claim or actions, and the undersigned will cooperate fully in any defense AAAI may make to such claim or action. Moreover, the undersigned agrees to cooperate in any claim or other action seeking to protect or enforce any right the undersigned has granted to AAAI in the article/paper. If any such claim or action fails because of facts that constitute a breach of any of the foregoing warranties, the undersigned agrees to reimburse whomever brings such claim or action for expenses and attorneys' fees incurred therein.

Returned Rights

In return for these rights, AAAI hereby grants to the above authors, and the employers for whom the work was performed, royalty-free permission to:

1. Retain all proprietary rights other than copyright (such as patent rights).
2. Personal reuse of all or portions of the above article/paper in other works of their own authorship.
3. Reproduce, or have reproduced, the above article/paper for the author's personal use, or for company use provided that AAAI copyright and the source are indicated, and that the copies are not used in a way that implies AAAI endorsement of a product or service of an employer, and that the copies per se are not offered for sale.
4. Make limited distribution of all or portions of the above article/paper prior to publication.
5. In the case of work performed under U.S. Government contract, AAAI grants the U.S. Government royalty-free permission to reproduce all or portions of the above article/paper, and to authorize others to do so, for U.S. Government purposes.

In the event the above article/paper is not accepted and published by AAAI, or is withdrawn by the author(s) before acceptance by AAAI, this agreement becomes null and void.

Author's Signature:_____

Date:_____

Employer for whom work was performed:_____

Title (if not author):_____

PART B-U.S. Government Employee Certification

This will certify that all authors of the above article/paper are employees of the U.S. Government and performed this work as part of their employment, and that the article/paper is therefore not subject to U.S. copyright protection.

The undersigned warrants that they are the sole author/translator of the above article/paper, and that the article/paper is original throughout, except for those portions shown to be in quotations.

U.S. Government Employee Authorized Signature:_____

Date:_____

Send to:

AAAI Press
445 Burgess Drive
Menlo Park, CA 94025

APPENDIX (Continued)

Publication Agreement

This is a publication agreement[1] ("this agreement") regarding a written manuscript currently entitled

---

("the article") to be published in PMLR ("the proceedings"). The parties to this Agreement are:

(name of corresponding author who signs on behalf of any other authors, collectively "you") and PMLR, ("the publisher").

1. By signing this form, you warrant that you are signing on behalf of all authors of the article, and that you have the authority to act as their agent for the purpose of entering into this agreement.
2. You hereby grant a Creative Commons copyright license in the article to the general public, in particular a Creative Commons Attribution 4.0 International License, which is incorporated herein by reference and is further specified at http://creativecommons.org/licenses/by/4.0/legalcode (human readable summary at http://creativecommons.org/licenses/by/4.0).
3. You agree to require that a citation to the original publication of the article in the proceedings as well as a hyperlink to the PMLR web site linking to the original paper be included in any attribution statement satisfying the attribution requirement of the Creative Commons license of paragraph 2.
4. You retain ownership of all rights under copyright in all versions of the article, and all rights not expressly granted in this agreement.
5. To the extent that any edits made by the publisher to make the article suitable for publication in the proceedings amount to copyrightable works of authorship, the publisher hereby assigns all right, title, and interest in such edits to you. The publisher agrees to verify with you any such edits that are substantive. You agree that the license of paragraph 2 covers such edits.

[1]The language of this publication agreement is based on Stuart Shieber's model open-access journal publication agreement, version 1.2, available at http://bit.ly/1m9UsNt.

APPENDIX (Continued)

6. You further warrant that:
    1. The article is original, has not been formally published in any other peerreviewed journal or in a book or edited collection, and is not under consideration for any such publication.
    2. You are the sole author(s) of the article, and that you have a complete and unencumbered right to make the grants you make.
    3. The article does not libel anyone, invade anyone's copyright or otherwise violate any statutory or common law right of anyone, and that you have made all reasonable efforts to ensure the accuracy of any factual information contained in the article. You agree to indemnify the publisher against any claim or action alleging facts which, if true, constitute a breach of any of the foregoing warranties or other provisions of this agreement, as well as against any related damages, losses, liabilities, and expenses incurred by the publisher.
7. This is the entire agreement between you and the publisher, and it may be modified only in writing. It will be governed by the laws of the Commonwealth of Massachusetts. It will bind and benefit our respective assigns and successors in interest, including your heirs. It will terminate if the publisher does not publish, in any medium, the article within one year of the date of your signature.

I HAVE READ AND AGREE FULLY WITH THE TERMS OF THIS AGREEMENT.

- Corresponding Author:
    - Signed:
    - Date:

CITED LITERATURE

1. Asif, K., Xing, W., Behpour, S., and Ziebart, B. D.: Adversarial cost-sensitive classification. In Proceedings of the Conference on Uncertainty in Artificial Intelligence, pages 92–101, 2015.

2. Wang, H., Xing, W., Asif, K., and Ziebart, B.: Adversarial prediction games for multivariate losses. In Advances in Neural Information Processing Systems, pages 2710–2718, 2015.

3. Behpour, S., Xing, W., and Ziebart, B. D.: ARC: Adversarial robust cuts for semi-supervised and multi-label classification. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 2704–2711, 2018.

4. Xing, W. and Ziebart, B.: Adversarial learning for 3d matching. In Conference on Uncertainty in Artificial Intelligence, pages 869–878. PMLR, 2020.

5. Fathony, R., Behpour, S., Zhang, X., and Ziebart, B.: Efficient and consistent adversarial bipartite matching. In International Conference on Machine Learning, pages 1456–1465, 2018.

6. Kolmogorov, V. and Zabin, R.: What energy functions can be minimized via graph cuts? IEEE transactions on pattern analysis and machine intelligence, 26(2):147–159, 2004.

7. Flake, G. W., Tarjan, R. E., and Tsioutsiouliklis, K.: Graph clustering and minimum cut trees. Internet Mathematics, 1(4):385–408, 2004.

8. Blum, A. and Chawla, S.: Learning from labeled and unlabeled data using graph mincuts. In International Conference on Machine Learning, pages 19–26. Morgan Kaufmann Publishers Inc., 2001.

9. Blum, A., Lafferty, J., Rwebangira, M. R., and Reddy, R.: Semi-supervised learning using randomized mincuts. In Proceedings of the twenty-first international conference on Machine learning, page 13. ACM, 2004.

10. Taskar, B., Chatalbashev, V., and Koller, D.: Learning associative markov networks. In Proceedings of the twenty-first international conference on Machine learning, page 102. ACM, 2004.

11. Boykov, Y., Veksler, O., and Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Transactions on pattern analysis and machine intelligence, 23(11):1222–1239, 2001.

12. Greig, D. M., Porteous, B. T., and Seheult, A. H.: Exact maximum a posteriori estimation for binary images. Journal of the Royal Statistical Society. Series B (Methodological), pages 271–279, 1989.

13. Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y.: Support vector machine learning for interdependent and structured output spaces. In Proceedings of the International Conference on Machine Learning, page 104. ACM, 2004.

14. Joachims, T.: A support vector method for multivariate performance measures. In Proceedings of the International Conference on Machine Learning, pages 377–384. ACM, 2005.

15. Kulesza, A. and Pereira, F.: Structured learning with approximate inference. In Advances in neural information processing systems, pages 785–792, 2008.

16. Lafferty, J., McCallum, A., and Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proc. of the International Conference on Machine Learning, pages 282–289, 2001.

17. Petterson, J., Yu, J., McAuley, J. J., and Caetano, T. S.: Exponential family graph matching and ranking. In Advances in Neural Information Processing Systems, pages 1455–1463, 2009.

18. Taskar, B., Chatalbashev, V., Koller, D., and Guestrin, C.: Learning structured prediction models: A large margin approach. In Proceedings of the International Conference on Machine Learning, pages 896–903. ACM, 2005.

19. Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y.: Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research, 6(Sep):1453–1484, 2005.

20. Tewari, A. and Bartlett, P.: On the consistency of multiclass classification methods. Journal of Machine Learning Research, 8:1007–1025, 2007.

21. Liu, Y.: Fisher consistency of multicategory support vector machines. In International Conference on Artificial Intelligence and Statistics, pages 291–298, 2007.

22. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y.: Generative adversarial nets. In Advances in Neural Information Processing Systems, pages 2672–2680, 2014.

23. Dalvi, N., Domingos, P., Sanghai, S., Verma, D., et al.: Adversarial classification. In KDD, pages 99–108. ACM, 2004.

24. Topsøe, F.: Information-theoretical optimization techniques. Kybernetika, 15(1):8–27, 1979.

25. Grünwald, P. D. and Dawid, A. P.: Game theory, maximum entropy, minimum discrepancy, and robust Bayesian decision theory. Annals of Statistics, 32:1367–1433, 2004.

26. Fisher, R. A.: On the mathematical foundations of theoretical statistics. Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character, 222(594-604):309–368, 1922.

27. Liu, A. and Ziebart, B. D.: Robust classification under sample selection bias. In Advances in Neural Information Processing Systems, pages 37–45, 2014.

28. Fathony, R., Liu, A., Asif, K., and Ziebart, B.: Adversarial multiclass classification: A risk minimization perspective. In Advances in Neural Information Processing Systems, pages 559–567, 2016.

29. Fathony, R., Bashiri, M. A., and Ziebart, B.: Adversarial surrogate losses for ordinal regression. In Advances in Neural Information Processing Systems, pages 563–573, 2017.

30. Li, J., Asif, K., Wang, H., Ziebart, B. D., and Berger-Wolf, T. Y.: Adversarial sequence tagging. In International Joint Conference on Artificial Intelligence, 2016.

31. Wainwright, M. J. and Jordan, M. I.: Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning, 1(1-2):1–305, 2008.

32. Kann, V.: Maximum bounded 3-dimensional matching is max snp-complete. Information Processing Letters, 37(1):27–35, 1991.

33. Potts, R. B.: Some generalized order-disorder transformations. In Mathematical proceedings of the cambridge philosophical society, volume 48, pages 106–109. Cambridge Univ Press, 1952.

34. Ishikawa, H.: Exact optimization for markov random fields with convex priors. IEEE transactions on pattern analysis and machine intelligence, 25(10):1333–1336, 2003.

35. Gao, W. and Zhou, Z.-H.: On the consistency of multi-label learning. In COLT, volume 19, pages 341–358, 2011.

36. Chan, P. K. and Stolfo, S. J.: Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In KDD, pages 164–168, 1998.

37. Elkan, C.: The foundations of cost-sensitive learning. In International joint conference on artificial intelligence, volume 17, pages 973–978. Lawrence Erlbaum Associates Ltd, 2001.

38. Zadrozny, B., Langford, J., and Abe, N.: Cost-sensitive learning by cost-proportionate example weighting. In ICDM, pages 435–442, 2003.

39. Zhou, Z.-H. and Liu, X.-Y.: On multi-class cost-sensitive learning. Computational Intelligence, 26(3):232–257, 2010.

40. Knoll, U., Nakhaeizadeh, G., and Tausend, B.: Cost-sensitive pruning of decision trees. In ECML, pages 383–386. Springer, 1994.

41. Turney, P. D.: Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. Journal of artificial intelligence research, pages 369–409, 1995.

42. Davis, J. V., Ha, J., Rossbach, C. J., Ramadan, H. E., and Witchel, E.: Cost-sensitive decision tree learning for forensic classification. In ECML, pages 622–629. Springer, 2006.

43. Brefeld, U., Geibel, P., and Wysotzki, F.: Support vector machines with example dependent costs. In ECML, pages 23–34. Springer, 2003.

44. Ling, C. X., Yang, Q., Wang, J., and Zhang, S.: Decision trees with minimal costs. In ICML, pages 544–551. ACM, 2004.

45. Lomax, S. and Vadera, S.: A survey of cost-sensitive decision tree induction algorithms. ACM Computing Surveys, 45(2):16, 2013.

46. Von Neumann, J. and Morgenstern, O.: Theory of games and economic behavior. Bull. Amer. Math. Soc, 51(7):498–504, 1945.

47. McMahan, H. B., Gordon, G. J., and Blum, A.: Planning in the presence of cost functions controlled by an adversary. In Proceedings of the International Conference on Machine Learning, pages 536–543, 2003.

48. Fathony, R., Rezaei, A., Bashiri, M. A., Zhang, X., and Ziebart, B.: Distributionally robust graphical models. In Advances in Neural Information Processing Systems, pages 8344–8355, 2018.

49. Fathony, R. Z. A.: Performance-Aligned Learning Algorithms with Statistical Guarantees. Doctoral dissertation, Université de Montréal, 2019.

50. Chen, J., Liu, Y., and Lu, S.: An improved parameterized algorithm for the minimum node multiway cut problem. Algorithmica, 55(1):1–13, 2009.

51. Călinescu, G., Karloff, H., and Rabani, Y.: An improved approximation algorithm for multiway cut. Journal of Computer and System Sciences, 60(3):564–574, 2000.

52. Naor, J. and Zosin, L.: A 2-approximation algorithm for the directed multiway cut problem. SIAM Journal on Computing, 31(2):477–482, 2001.

53. Boyd, S. and Vandenberghe, L.: Convex optimization. Cambridge university press, 2004.

54. von Neumann, J. and Morgenstern, O.: Theory of Games and Economic Behavior. Princeton University Press, 1947.

55. Finley, T. and Joachims, T.: Training structural svms when exact inference is intractable. In Proceedings of the 25th international conference on Machine learning, pages 304–311. ACM, 2008.

56. Duchi, J., Hazan, E., and Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 12(Jul):2121–2159, 2011.

57. Zhang, L., Li, Y., and Nevatia, R.: Global data association for multi-object tracking using network flows. In CVPR, pages 1–8. IEEE, 2008.

58. Chari, V., Lacoste-Julien, S., Laptev, I., and Sivic, J.: On pairwise costs for network flow multi-object tracking. In CVPR, pages 5537–5545, 2015.

59. Tang, S., Andres, B., Andriluka, M., and Schiele, B.: Subgraph decomposition for multi-target tracking. In CVPR, pages 5033–5041, 2015.

60. Keuper, M., Tang, S., Zhongjie, Y., Andres, B., Brox, T., and Schiele, B.: A multi-cut formulation for joint segmentation and tracking of multiple objects. arXiv preprint arXiv:1607.06317, 2016.

61. Tang, S., Andriluka, M., Andres, B., and Schiele, B.: Multiple people tracking by lifted multicut and person re-identification. In CVPR, pages 3539–3548, 2017.

62. Sion, M.: On general minimax theorems. Pacific Journal of mathematics, 8(1):171–176, 1958.

63. Papadimitriou, C. H.: On the complexity of integer programming. Journal of the ACM (JACM), 28(4):765–768, 1981.

64. Schmidt, M., Berg, E., Friedlander, M., and Murphy, K.: Optimizing costly functions with simple constraints: A limited-memory projected quasi-Newton algorithm. In Artificial Intelligence and Statistics, pages 456–463, 2009.

65. Douglas, J. and Rachford, H. H.: On the numerical solution of heat conduction problems in two and three space variables. Transactions of the American Mathematical Society, 82(2):421–439, 1956.

66. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends® in Machine Learning, 3(1):1–122, 2011.

67. Liu, L. and Han, Z.: Multi-block admm for big data optimization in smart grid. In 2015 International Conference on Computing, Networking and Communications (ICNC), pages 556–561. IEEE, 2015.

68. Duchi, J., Shalev-Shwartz, S., Singer, Y., and Chandra, T.: Efficient projections onto the l 1-ball for learning in high dimensions. In Proceedings of the International Conference on Machine Learning, pages 272–279. ACM, 2008.

69. Birkhoff, G.: Three observations on linear algebra. Univ. Nac. Tacuman, Rev. Ser. A, 5:147–151, 1946.

70. Von Neumann, J.: A certain zero-sum two-person game equivalent to the optimal assignment problem. Contributions to the Theory of Games, 2:5–12, 1953.

71. Cui, L.-B., Li, W., and Ng, M. K.: Birkhoff–von neumann theorem for multistochastic tensors. SIAM Journal on Matrix Analysis and Applications, 35(3):956–973, 2014.

72. Del Pia, A., Dey, S. S., and Molinaro, M.: Mixed-integer quadratic programming is in np. Mathematical Programming, 162(1-2):225–240, 2017.

73. Premebida, C., Monteiro, G., Nunes, U., and Peixoto, P.: A lidar and vision-based approach for pedestrian and vehicle detection and tracking. In 2007 IEEE intelligent transportation systems conference, pages 1044–1049. IEEE, 2007.

74. Dua, D. and Graff, C.: UCI machine learning repository, 2017.

75. Dias, D. B., Madeo, R. C., Rocha, T., Bíscaro, H. H., and Peres, S. M.: Hand movement recognition for brazilian sign language: a study using distance-based neural networks. In 2009 international joint conference on neural networks, pages 697–704. IEEE, 2009.

76. Altman, N. S.: An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician, 46(3):175–185, 1992.

77. Escalera, S., Pujol, O., and Radeva, P.: On the decoding process in ternary error-correcting output codes. IEEE transactions on pattern analysis and machine intelligence, 32(1):120–134, 2008.

78. Leal-Taixé, L., Milan, A., Reid, I., Roth, S., and Schindler, K.: Motchallenge 2015: Towards a benchmark for multi-target tracking. arXiv preprint arXiv:1504.01942, 2015.

79. Kim, S., Kwak, S., Feyereisl, J., and Han, B.: Online multi-target tracking by large margin structured learning. In Asian Conference on Computer Vision, pages 98–111. Springer, 2012.

80. He, D.-C. and Wang, L.: Texture unit, texture spectrum, and texture analysis. IEEE transactions on Geoscience and Remote Sensing, 28(4):509–512, 1990.

81. Beauchemin, S. S. and Barron, J. L.: The computation of optical flow. ACM Computing Surveys (CSUR), 27(3):433–466, 1995.

82. Joachims, T.: SVM-struct: Support vector machine for complex outputs. `http://www.cs.cornell.edu/People/tj/svm_light/svm_struct.html`, 2008.

83. Vedaldi, A.: A MATLAB wrapper of SVM$^{\text{struct}}$. `http://www.vlfeat.org/~vedaldi/code/svm-struct-matlab.`, 2011.

84. Schmidt, M.: minConf: projection methods for optimization with simple constraints in Matlab. `http://www.cs.ubc.ca/~schmidtm/Software/minConf.html`, 2008.

VITA

NAME Wei Xing

EDUCATION B.S., Information and Computing Science, Huazhong University of Science and Technology, Wuhan, China, 2010

EDUCATION M.S., Computer Science, Zhejiang University, Zhejiang, China, 2013

PUBLICATIONS Kaiser Asif, Wei Xing, Sima Behpour, Brian D Ziebart. "Kaiser Asif, Wei Xing, Sima Behpour, Brian D Ziebart" In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI, 2015)

Hong Wang, Wei Xing, Kaiser Asif, Brian D Ziebart. "Adversarial Prediction Games for Multivariate Losses" In Proceedings of the Advances in Neural Information Processing Systems (NIPS, 2015)

Sima Behpour, Wei Xing, Brian D. Ziebart. "ARC: Adversarial Robust Cuts for Semi-Supervised and Multi-Label Classification" In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (IEEE/CVF, 2018)

Rizal Fathony, Kaiser Asif, Anqi Liu, Mohammad Ali Bashiri, Wei Xing, Sima Behpour, Xinhua Zhang, Brian D Ziebart. "Consistent robust adversarial prediction for general multiclass classification" arXiv preprint arXiv:1812.07526 (2018)

Wei Xing, Brian D Ziebart. "Adversarial Learning for 3D Matching" In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI, 2020)