Adversarial Learning for 3D Matching

Wei Xing
Computer Science Department
University of Illinois at Chicago
Chicago, IL 60607

Brian D. Ziebart
Computer Science Department
University of Illinois at Chicago
Chicago, IL 60607

Abstract

Structured prediction of objects in spaces that are inherently difficult to search or compactly characterize is a particularly challenging task. For example, though bipartite matchings in two dimensions can be tractably optimized and learned, the higher-dimensional generalization—3D matchings—are NP-hard to optimally obtain and the set of potential solutions cannot be compactly characterized. Though approximation is therefore necessary, prevalent structured prediction methods inherit the weaknesses they possess in the two-dimensional setting—either suffering from inconsistency or intractability—even when the approximations are sufficient. In this paper, we explore extending an adversarial approach to learning bipartite matchings that avoids these weaknesses to the three dimensional setting. We assess the benefits compared to margin-based methods on a three-frame tracking problem.

1 INTRODUCTION

Machine learning for complex structured data and interrelated variables is increasingly important for applications in computer vision, natural language processing, computational biology, and other areas. Among those learning tasks, some of them have certain restricted relationships and structures (e.g., chains, trees, and other low-treewidth structures) that facilitate efficient inference algorithms. For example, binary-valued associative Markov networks [Taskar et al., 2004] and the special case of attractive pairwise relationships [Boykov et al., 2001], use minimum graph cuts [Greig et al., 1989] for inference and maximum margin methods [Joachims, 2005] for training.

Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI), PMLR volume 124, 2020.

Unfortunately, many tasks with structured data are intractable and approximation methods are required for practical uses. Learning methods that provide strong guarantees when using exact solutions often lose those theoretical guarantees and desirable properties when we use these approximation methods. Indeed, learning can fail even when using an approximate inference method with rigorous approximation guarantees [Kulesza and Pereira, 2008]. First, approximation methods can effectively reduce the expressivity of an underlying model by making it impossible to choose parameters that reliably give good predictions. Second, approximations can respond to parameter changes in such a way that standard learning algorithms are misled. Even without involving approximations, many existed method have some drawbacks when using exact inference. For problems like bipartite matchings, the exponentiated potential fields models [Lafferty et al., 2001, Petterson et al., 2009] become intractable to normalize as set sizes grows. Meanwhile, maximum margin methods such as Structured Support Vector Machine (SSVM) [Taskar et al., 2005, Tsochantaridis et al., 2005] do not have Fisher consistency [Tewari and Bartlett, 2007, Liu, 2007].

Adversarial learning has been proposed to increase the robustness of learned models [Dalvi et al., 2004] with recent applications to generative models [Goodfellow et al., 2014]. We consider the supervised setting with adversarial uncertainty in this work. Unlike SSVM and some other methods that use hinge loss surrogate that can be quite loose in practice. Adversarial learning provides an adaptive way to reduce the 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. This provides Fisher consistency [Fisher, 1922] on a wide range of loss functions, which guarantees the model converges to make loss minimizing predictions given more and more training samples. All these great properties lead to the success of adversarial methods on problems such as cost-sensitive classification [Asif et al., 2015], classification under covariate shift [Liu and Ziebart, 2014], classification problems with zero-one loss [Fathony et al., 2016], ordinal regression [Fathony et al., 2017] and chain structures [Li et al., 2016].

Following and extending research that uses this minimax perspective for structured prediction to learn bipartite matchings [Fathony et al., 2018], this paper addresses the problem of learning three-dimensional (3D) matchings. 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 [Kann, 1991]. 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.

2 BACKGROUND

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) No 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 1 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.

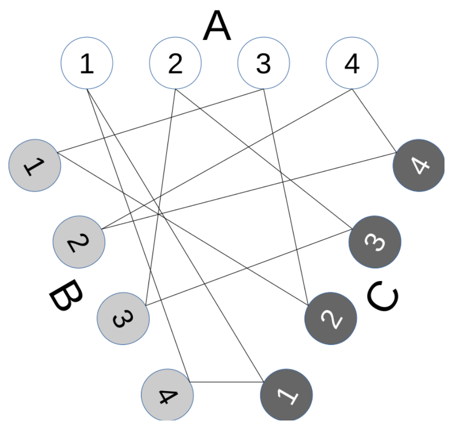

Figure 1: $n = 4$ 3D matching with matches $\{141, 233, 312, 424\}$

Unlike weighted bipartite matching, which can be solved in polynomial time using Hungarian method [Kuhn, 1955], weighted maximum 3D matching problems or the balanced 3D assignment problem is NP-hard (generalizing the NP-hard 0-1 version [Kann, 1991]).

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 artificial for this domain, this prevents existing successful methods based on min-cost flow network algorithms [Zhang et al., 2008, Chari et al., 2015, Tang et al., 2015, Keuper et al., 2016, Tang et al., 2017] from being applicable.

As a learning task, we need to solve the inverse problem of learning parameters that for potentials ψ so that the 3D matchings in training data are indeed the maximum matchings. It is useful for the potential function to include information from elements in the three sets in the form of $\psi(\pi_1, \pi_2, \mathbf{x})$. Here $\mathbf{x}$ is a general representation of all the raw data in all the three sets whose elements values lie in $\boldsymbol{\mathcal{X}}$. Depending on the information, $\mathbf{x}$ can be a vector or even high dimensional tensor. We denote its corresponding random variable as $\mathbf{X}$. For permutation π, we similarly denote its random variable as Π. Now, given the training dataset, we can assume they follow an empirical distribution $\tilde{P}(\mathbf{X}, \Pi)$.

In this paper, an important assumption is that the potential ψ is a linear function of a vector of parameters $\boldsymbol{\theta}$

inner product a vector of features $\boldsymbol{\phi}$: $\psi = \boldsymbol{\theta} \cdot \boldsymbol{\phi}$. The features $\boldsymbol{\phi}$ is decided by the matches and the matched elements. For general problems, let's use y instead of π as the labels, and then we can 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, $\mathbf{y}_{\mathbf{c}} = \{y_{c_1}, \ldots, y_{c_l}\}$ is the corresponding set of label values for the variables in the subset. In the particular problem of 3D matching, we have the form $\boldsymbol{\phi}(\pi_1, \pi_2, \mathbf{x})$

2.2 MARKOV NETWORKS

The Markov network is one of the general-purpose state-of-art approaches for structured prediction. It can be treated as a log-linear model when its density is positive. A Markov network can be written as:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} e^{\boldsymbol{\psi}(\mathbf{y},\mathbf{x})}. \quad (1)$$

The structure of the potential ψ is an undirected graphical model and the variables in set $\mathbf{c}$ form cliques that are connected by undirected edges

In most cases, only pairwise and unary potential functions are used. However, even with this limitation, the most probable assignment of values, $\mathbf{y}^* = \operatorname{argmax}_{\mathbf{y} \in \boldsymbol{\mathcal{Y}}} P(\mathbf{y}|\mathbf{x})$, and the normalization term, $Z(\mathbf{x}) = \sum_{\mathbf{y} \in \boldsymbol{\mathcal{Y}}} e^{\sum_{\mathbf{c} \in C} \boldsymbol{\psi}_{\mathbf{c}}(\mathbf{y}_{\mathbf{c}}, \mathbf{x})}$, are both still intractable in general [Wainwright and Jordan, 2008]. Often undirected graphs with low tree-width (e.g., chains, trees) are employed so that efficient maximization and normalization computations can be achieved [Wainwright and Jordan, 2008].

2.3 MAXIMUM MARGIN LEARNING

Maximum margin methods for learning are another option for structured prediction. The structured support vector machine (SSVM) [Tsochantaridis et al., 2004, Taskar et al., 2005] optimizes over a hinge loss surrogate of the original loss, which can be generally non-convex and possibly not even continuous. It takes the following form:

$$\min_{\boldsymbol{\theta}, \boldsymbol{\epsilon} \geq \mathbf{0}} ||\boldsymbol{\theta}|| + \lambda \sum_i \epsilon_i \text{ s.t. :}$$
$$\epsilon_i \geq \max_{\mathbf{y}'} \text{loss}(\mathbf{y}', \mathbf{y}^{(i)}) + \psi(\mathbf{y}', \mathbf{x}) - \psi(\mathbf{y}^{(i)}, \mathbf{x}), \quad (2)$$

where loss() is any target loss function, $\boldsymbol{\theta}$ is the potential function parameter, λ is a pre-set regularization parameter, and ϵ_i is the hinge loss incurred by the i^{th} training example.

Unfortunately, the hinge loss can be quite loose. In some cases, for a particular example $\mathbf{y}^{(i)}$, it may be far larger than the actual loss, $\text{loss}(\hat{\mathbf{y}}, \mathbf{y}^{(i)})$, and larger than random guessing or the worst possible loss, $\max_{\mathbf{y}} \text{loss}(\mathbf{y}, \mathbf{y}^{(i)})$[Behpour et al., 2018]. In such cases, the hinge loss bounds do not provide meaningful guarantees on the predictor's performance.

2.4 ADVERSARIAL LEARNING

Adversarial learning aims to compete with a worst-case distribution approximating the training labels under certain constraints preserving the properties of training data [Topsøe, 1979, Grünwald and Dawid, 2004, Asif et al., 2015]. Let $\check{P}(\check{\mathbf{Y}}|\mathbf{X})$ be the adversary's mixed strategy to generate worse-case labels, i.e., maximize the loss, while the predictor's mixed strategy is $\hat{P}(\hat{\mathbf{Y}}|\mathbf{X})$, aiming to minimize the loss. $\tilde{P}$ is the empirical distribution. We can formalize any general learning problem in this way:

$$\min_{\hat{P}(\hat{\mathbf{Y}}|\mathbf{X}) \in \Delta} \max_{\check{P}(\check{\mathbf{Y}}|\mathbf{X}) \in \Delta \cap \Xi} \mathbb{E}_{\substack{\mathbf{X} \sim \tilde{P}; \hat{\mathbf{Y}}|\mathbf{X} \sim \hat{P}; \\ \check{\mathbf{Y}}|\mathbf{X} \sim \check{P}}}, \left[\text{loss}(\hat{\mathbf{Y}}, \check{\mathbf{Y}}) \right], \quad (3)$$

with Δ representing 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.

One important difference between adversarial learning and other methods is that it no longer directly learns from training samples, but instead learned by optimizing against the adversary's approximation of training labels. Note that the adversary's choice is not static and instead depends on the predictor's mixed strategy. In this model, the adversary tries its best to make the predicted label as uncertain as possible. Without constraint Ξ, nothing could be learned. However, when proper Ξ is chosen, such as the statistics from the training data, the adversary can be forced to become highly predictable.

In exchange for not optimizing over the exact training samples, this method is able to train based on the exact loss (e.g., 0-1 loss for classification or Hamming loss for structured prediction) and still forms a convex optimization problem, hence no longer needing to use explicit surrogate losses. In fact, the training error is always upper bounded by the game value. Optimizing over the game matrix can more closely bound the actual loss [Asif et al., 2015].

For a polynomial-sized game matrix, the problem can be solved directly efficiently [Asif et al., 2015]. However, for exponentially-sized game matrices (e.g., from complex problems involving structured losses, such as F-measure, or structured relationships between predicted variables) explicit formulations are intractable, and constraint generation methods, such as double oracle[McMahan et al., 2003] are employed to gradually increase the size of the game matrix and reach the final equilibrium.

3 APPROACH

3.1 MINIMAX GAME FORMULATION

Following the adversarial approach applied to the task of learning bipartite matchings/permutations [Fathony et al., 2018], we can build the following objective function for the task of learning 3D matchings:

$$\min_{\hat{P}(\hat{\Pi}_1,\hat{\Pi}_2|\mathbf{X})} \max_{\check{P}(\check{\Pi}_1,\check{\Pi}_2|\mathbf{X})} \mathbb{E}_{\mathbf{X}\sim\tilde{P};\hat{\Pi}_1,\hat{\Pi}_2|\mathbf{X}\sim\hat{P};\check{\Pi}_1,\check{\Pi}_2|\mathbf{X}\sim\check{P}} \left[\mathrm{loss}(\hat{\Pi}_1,\hat{\Pi}_2,\check{\Pi}_1,\check{\Pi}_2)\right]$$
$$\text{s.t. } \mathbb{E}_{\mathbf{X}\sim\tilde{P},\check{\Pi}_1,\check{\Pi}_2|\mathbf{X}\sim\check{P}} \left[\sum_{i=1}^{n} \phi_i(\check{\Pi}_{1,i},\check{\Pi}_{2,i},\mathbf{X})\right] = \mathbb{E}_{(\mathbf{X},\Pi_1,\Pi_2)\sim\tilde{P}} \left[\sum_{i=1}^{n} \phi_i(\Pi_{1,i},\Pi_{2,i},\mathbf{X})\right]. \quad (4)$$

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 [Von Neumann and Morgenstern, 1945, Sion, 1958], the optimization in Eq. (4) can be equivalently solved in the dual formulation:

$$\min_{\theta} \mathbb{E}_{\mathbf{x},\Pi_1,\Pi_2\sim\tilde{P}} \min_{\hat{P}(\hat{\Pi}_1,\hat{\Pi}_2|\mathbf{x})} \max_{\check{P}(\check{\Pi}_1,\check{\Pi}_2|\mathbf{x})} \mathbb{E}_{\substack{\hat{\Pi}_1,\hat{\Pi}_2|\mathbf{x}\sim\hat{P} \\ \check{\Pi}_1,\check{\Pi}_2|\mathbf{x}\sim\check{P}}} \Bigg[\mathrm{loss}(\hat{\Pi}_1,\hat{\Pi}_2,\check{\Pi}_1,\check{\Pi}_2) + \theta \cdot \sum_{i=1}^{n} \Big(\phi_i(\check{\Pi}_{1,i},\check{\Pi}_{2,i},\mathbf{x}) - \phi_i(\Pi_{1,i},\Pi_{2,i},\mathbf{x}) \Big) \Bigg], \quad (5)$$

where θ is the Lagrange dual variable for the moment matching constraints. For this problem, we use the match-based Hamming distance, $\mathrm{loss}(\hat{\pi}_1,\hat{\pi}_2,\check{\pi}_1,\check{\pi}_2) = \frac{1}{n}\sum_{i=1}^{n} 1_{\hat{\pi}_{1,i}\neq\check{\pi}_{1,i}}(\hat{\pi}_{1,i},\check{\pi}_{1,i}) \vee 1_{\hat{\pi}_{2,i}\neq\check{\pi}_{2,i}}(\hat{\pi}_{2,i},\check{\pi}_{2,i})$, 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 $\check{\mathbf{p}}_{\mathbf{X}}, \hat{\mathbf{p}}_{\mathbf{X}} \in \boldsymbol{\Delta}$, which is the vector of conditional probabilities of all the possible $\hat{y}$ or $\check{y}$ given x. Then we can rewrite (5) in matrix form as:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x},\Pi_1,\Pi_2\sim\tilde{P}} \left[\left(\max_{\check{\mathbf{p}}_{\mathbf{X}}} \min_{\hat{\mathbf{p}}_{\mathbf{X}}} \hat{\mathbf{p}}_{\mathbf{X}}^{\mathrm{T}} \, \mathbf{C}_{\boldsymbol{\theta},\mathbf{X}} \, \check{\mathbf{p}}_{\mathbf{X}} \right) - \sum_{i} \boldsymbol{\theta} \cdot \boldsymbol{\phi}(\Pi_{1,i},\Pi_{2,i},\mathbf{X}) \right]. \quad (6)$$

where $(\mathbf{C}_{\theta,\mathbf{X}})_{\hat{y},\check{y}} = \mathbf{C}_{\hat{y},\check{y}} + \theta^{\mathrm{T}}(\phi(\check{y},x) - \phi(\tilde{y},x))$.

Table 1 is the payoff matrix $\mathbf{C}_{\boldsymbol{\theta},\mathbf{X}}$ 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 1: Augmented Hamming loss matrix, n=3

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

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.

3.2 MARGINAL DISTRIBUTION FORMULATION

For this problem, using the double oracle method leads to solving 3D matching problems multiple times. Following [Fathony et al., 2018], 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.

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. 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 $\phi_i(x,j,k)$. 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 $\check{P}$).

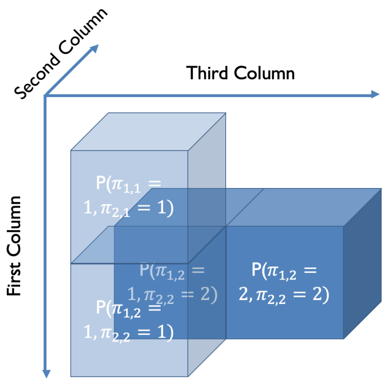
Second Column
Third Column
First Column
P($\pi_{1,1}$ = 1, $\pi_{2,1}$ = 1)
P($\pi_{1,2}$ = 1, $\pi_{2,2}$ = 2)
P($\pi_{1,2}$ = 2, $\pi_{2,2}$ = 2)
P($\pi_{1,2}$ = 1, $\pi_{2,2}$ = 1)

Figure 2: 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. (5) can be rewrite as a minimax over marginal probability tensors $\mathbf{P}$ and $\mathbf{Q}$. The constraints are also reformed, and we have:

$$\begin{aligned} &\min_{\theta} \mathbb{E}_{\mathbf{X},\mathbf{Y}\sim\tilde{P}} \min_{\mathbf{P}\geq\mathbf{0}} \max_{\mathbf{Q}\geq\mathbf{0}} \Big[1 - \frac{1}{n}\langle \mathbf{P}, \mathbf{Q}\rangle \\ &\qquad + \langle \mathbf{Q} - \mathbf{Y}, \textstyle\sum_o \theta_o \mathbf{X}_o\rangle\Big] \\ \text{s.t. :} &\sum_{i,j} \mathbf{P}_{i,j,k} = \sum_{i,j} \mathbf{Q}_{i,j,k} = 1, \forall k \in [n] \\ &\sum_{j,k} \mathbf{P}_{i,j,k} = \sum_{j,k} \mathbf{Q}_{i,j,k} = 1, \forall i \in [n] \\ &\sum_{i,k} \mathbf{P}_{i,j,k} = \sum_{i,k} \mathbf{Q}_{i,j,k} = 1, \forall j \in [n], \end{aligned} \tag{7}$$

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. (7) as hyperplanar stochastic tensors.

For bipartite matching, the Birkhoff–von Neumann theorem [Birkhoff, 1946, Von Neumann, 1953] 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. However, this result does not generalize to 3D matching. Some extreme points of the multistochastic tensor are not convex combinations of permutation tensors [Cui et al., 2014]. Though this tensor differs from ours, this implies that our marginal formulation is a relaxation from the original mixed strategy of permutations.

3.2.1 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 $\mathbf{Q}$ 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 $\mathbf{P}$ and $\mathbf{Q}$ as well as a regularization penalty to the parameter θ to prevent overfitting in our model. Also the empirical expectation in Eq. (7) can be replaced by the average over training samples. Then we have the following optimization:

$$\begin{aligned} &\max_{\mathbf{Q}\geq\mathbf{0}} \min_{\theta} \frac{1}{m} \sum_{l=1}^{m} \min_{\mathbf{P}^{(l)}\geq\mathbf{0}} \Big[\Big\langle \mathbf{Q}^{(l)} \\ &\qquad - \mathbf{Y}^{(l)}, \textstyle\sum_o \theta_o \mathbf{X}_o^{(l)} \Big\rangle - \frac{1}{n}\Big\langle \mathbf{P}^{(l)}, \mathbf{Q}^{(l)}\Big\rangle \\ &\qquad + \tfrac{\mu}{2}\|\mathbf{P}^{(l)}\|_F^2 - \tfrac{\mu}{2}\|\mathbf{Q}^{(l)}\|_F^2\Big] + \tfrac{\lambda}{2}\|\theta\|_2^2 \\ \text{s.t. :} &\sum_{i,j} \mathbf{P}^{(l)}_{i,j,k} = \sum_{i,j} \mathbf{Q}^{(l)}_{i,j,k} = 1, \forall k \in [n] \\ &\sum_{j,k} \mathbf{P}^{(l)}_{i,j,k} = \sum_{j,k} \mathbf{Q}^{(l)}_{i,j,k} = 1, \forall i \in [n] \\ &\sum_{i,k} \mathbf{P}^{(l)}_{i,j,k} = \sum_{i,k} \mathbf{Q}^{(l)}_{i,j,k} = 1, \forall j \in [n], \end{aligned} \tag{8}$$

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 $\|A\|_F$ denotes the Frobenius norm of tensor 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. 8, 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:

$$\theta_k^* = -\frac{1}{\lambda m} \sum_{l=1}^{m} \Big\langle \mathbf{Q}^{(l)} - \mathbf{Y}^{(l)}, \mathbf{X}_o^{(l)} \Big\rangle. \tag{9}$$

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

$$\begin{aligned}\mathbf{P}^{(l)*} &= \underset{\{\mathbf{P}^{(l)}\geq\mathbf{0}\}}{\operatorname{argmin}} \tfrac{\mu}{2}\|\mathbf{P}^{(l)}\|_F^2 - \frac{1}{n}\left\langle \mathbf{P}^{(l)}, \mathbf{Q}^{(l)} \right\rangle \\ &= \underset{\{\mathbf{P}^{(l)}\geq\mathbf{0}\}}{\operatorname{argmin}} \|\mathbf{P}^{(l)} - \tfrac{1}{n\mu}\mathbf{Q}^{(l)}\|_F^2 \\ \text{s.t.}: &\sum_{i,j}\mathbf{P}^{(l)}_{i,j,k} = 1, \forall k\in[n]; \sum_{j,k}\mathbf{P}^{(l)}_{i,j,k} = 1, \forall i \in [n] \\ &\sum_{i,k}\mathbf{P}^{(l)}_{i,j,k} = 1, \forall j\in[n]. \end{aligned} \tag{10}$$

This minimization is equivalent to projecting the matrix $\frac{1}{n\mu}\mathbf{Q}^{(l)}$ 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 [Schmidt et al., 2009] to find the best $\mathbf{Q}$. After we achieve the adversary's optimal marginal probability $\mathbf{Q}^*$, we can use Eq. (9) to get θ^*, which is used in the prediction step.

3.2.2 Hyperplanar Stochastic Tensors Projection

The projection from an arbitrary tensor $\mathbf{R}$ to the set of hyperplanar stochastic tensors can be formulated as:

$$\begin{aligned} &\min_{\mathbf{P}\geq\mathbf{0}} \|\mathbf{P}-\mathbf{R}\|_F^2, \\ \text{s.t.}: &\sum_{i,j}\mathbf{P}_{i,j,k} = 1, \forall k\in[n]; \sum_{j,k}\mathbf{P}_{i,j,k} = 1, \forall i\in[n] \\ &\sum_{i,k}\mathbf{P}_{i,j,k} = 1, \forall j\in[n]. \end{aligned} \tag{11}$$

The alternating direction method of multipliers (ADMM) technique [Douglas and Rachford, 1956, Boyd et al., 2011] is a powerful tool for solving this problem. 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:

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

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 [Liu and Han, 2015].

The hyperplanar stochastic tensors constraint can be divided 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. 11 can be rewrite as:

$$\begin{aligned} \min_{\mathbf{P},\mathbf{S},\mathbf{T}} &\tfrac{1}{2}\|\mathbf{P}-\mathbf{R}\|_F^2 + \tfrac{1}{2}\|\mathbf{S}-\mathbf{R}\|_F^2 + \tfrac{1}{2}\|\mathbf{T}-\mathbf{R}\|_F^2 + \\ &\mathrm{I}_{C_1}(\mathbf{P}) + \mathrm{I}_{C_2}(\mathbf{S}) + \mathrm{I}_{C_3}(\mathbf{T}) \\ &\text{s.t.}: \mathbf{P}-\mathbf{S} = 0 \text{ and } \mathbf{P}-\mathbf{T} = 0, \end{aligned} \tag{12}$$

where $\mathrm{I}_C(x)$ 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:

$$\begin{aligned} \mathcal{L}_\rho(\mathbf{P},\mathbf{S},\mathbf{T},\mathbf{W_1},\mathbf{W_2}) = &\tfrac{1}{2}\|\mathbf{P}-\mathbf{R}\|_F^2 + \tfrac{1}{2}\|\mathbf{S}-\mathbf{R}\|_F^2 \\ &+\tfrac{1}{2}\|\mathbf{T}-\mathbf{R}\|_F^2 + \mathrm{I}_{C_1}(\mathbf{P}) + \mathrm{I}_{C_2}(\mathbf{S}) \\ &+\mathrm{I}_{C_3}(\mathbf{T}) + \tfrac{\rho}{2}\|\mathbf{P}-\mathbf{S}+\mathbf{W_1}\|_F^2 \\ &+\tfrac{\rho}{2}\|\mathbf{P}-\mathbf{T}+\mathbf{W_2}\|_F^2 \end{aligned} \tag{13}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are the scaled dual variable. From the above formula, we can compute the update for $\mathbf{P}$ as:

$$\begin{aligned} \mathbf{P}^{t+1} &= \underset{\mathbf{P}}{\operatorname{argmin}} \mathcal{L}_\rho(\mathbf{P},\mathbf{S}^t,\mathbf{T}^t,\mathbf{W}_1^t,\mathbf{W}_2^t) \\ &= \underset{\{\mathbf{P}|\mathbf{P}\in C_1\}}{\operatorname{argmin}} \tfrac{1}{2}\|\mathbf{P}-\mathbf{R}\|_F^2 + \tfrac{\rho}{2}\|\mathbf{P}-\mathbf{S}^t+\mathbf{W}_1^t\|_F^2 \\ &\quad+\tfrac{\rho}{2}\|\mathbf{P}-\mathbf{T}^t+\mathbf{W}_2^t\|_F^2 \\ &= \underset{\{\mathbf{P}|\mathbf{P}\in C_1\}}{\operatorname{argmin}} \|\mathbf{P} - \tfrac{1}{1+2\rho}\left(\mathbf{R} + \rho\left(\mathbf{S}^t + \mathbf{T}^t \right.\right. \\ &\quad\left.\left. -\mathbf{W_1}^t - \mathbf{W_2}^t\right)\right)\|_F^2. \end{aligned} \tag{14}$$

This is actually a projection to the set C_1 by projecting to the probability simplex independently for each slice of the tensor $\frac{1}{1+2\rho}\left(\mathbf{R} + \rho\left(\mathbf{S}^t + \mathbf{T}^t - \mathbf{W}_1^t - \mathbf{W}_2^t\right)\right)$. 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 [Duchi et al., 2008]. We list all the update strategy here:

$$\mathbf{P}^{t+1} = \mathrm{Proj}_{C_1}\left(\tfrac{1}{1+2\rho}\left(\mathbf{R}+\rho\left(\mathbf{S}^t+\mathbf{T}^t-\mathbf{W}_1^t-\mathbf{W}_2^t\right)\right)\right)$$

$$\mathbf{S}^{t+1} = \mathrm{Proj}_{C_2}\left(\tfrac{1}{1+\rho}\left(\mathbf{R}+\rho\left(\mathbf{P}^{t+1}+\mathbf{W}_1^t\right)\right)\right) \tag{15}$$

$$\mathbf{T}^{t+1} = \mathrm{Proj}_{C_3}\left(\tfrac{1}{1+\rho}\left(\mathbf{R}+\rho\left(\mathbf{P}^{t+1}+\mathbf{W}_2^t\right)\right)\right) \tag{16}$$

$$\mathbf{W}_1^{t+1} = \mathbf{W}_1^t + \mathbf{P}^{t+1} - \mathbf{S}^{t+1} \tag{17}$$

$$\mathbf{W}_2^{t+1} = \mathbf{W}_2^t + \mathbf{P}^{t+1} - \mathbf{T}^{t+1}. \tag{18}$$

These updating steps are repeated until the primal and dual residual optimality is reached [Boyd et al., 2011].

3.2.3 Prediction

In the prediction step, we first use the θ^* we learned and the testing data to solve the inner optimization problem in Eq. 10, 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.

We pursue the following problem:

$$\begin{aligned} &\underset{\mathbf{Y}}{\operatorname{argmin}} \ \|\mathbf{P}-\mathbf{Y}\|_F^2 \\ \text{s.t. :} &\sum_{i,j}\mathbf{Y}_{i,j,k}=1, \forall k \in [n]; \sum_{j,k}\mathbf{Y}_{i,j,k}=1, \forall i \in [n]; \\ &\sum_{i,k}\mathbf{Y}_{i,j,k}=1, \forall j \in [n]; \mathbf{Y}_{i,j,k}\in\{0,1\} \end{aligned} \quad (19)$$

$\mathbf{Y}$ is prediction we want and the whole problem is a integer quadratic programming problem. This problem is NP-hard [Del Pia et al., 2017], but there are approximation algorithms that can often be used for solving it in practice.

4 EXPERIMENT AND EVALUATION

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 [Leal-Taixé et al., 2015].

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 2 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.

Table 2: 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

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.

4.1 FEATURE REPRESENTATION

The synthetic data, the features $\phi_i(\pi_{1,i}, \pi_{2,i}, X)$ contain: a copy of the raw data, summation of each pair of raw data in the given tuple, the overall summation, the absolute difference between raw data, maximal and minimal L2 norm value of the raw data and group differences.

For the video tracking problem, we use an existing feature representation [Kim et al., 2012] that uses six different types of features: intersection over union (IoU) overlap ratio between bounding boxes, Euclidean distance between object centers, 21 color histogram distance features (RGB) from the Bhattacharyaa distance, 21 local binary pattern (LBP) features from similar Bhattacharyaa distances and bounding box blocks, Optical flow (motion) between bounding boxes and Four indicator variables (for *entering*, *leaving*, *hiding in the middle* and *staying invisible).*

4.2 EXPERIMENT SETUP

To make the comparison with our method adversarial 3D matching (Adv3DMarg), we implement the SSVM model [Taskar et al., 2005, Tsochantaridis et al., 2005] based on [Kim et al., 2012] using `SVM-Struct` [Joachims, 2008, Vedaldi, 2011] and two-stage marginal

Table 3: The mean and standard deviation (in parenthesis) of the average accuracy for synthetic data.

# Objects	2S-Adv Marg.	Adv3D Marg.	SSVM
3	0.852 (0.09)	0.885 (0.08)	0.893 (0.09)
4	0.812 (0.10)	**0.855** (0.10)	0.833 (0.09)
5	0.815 (0.12)	**0.827** (0.11)	0.802 (0.10)
6	0.779 (0.10)	**0.808** (0.11)	0.800 (0.09)

Table 4: The mean and standard deviation (in parenthesis) of the average accuracy for video tracking.

Training/ Testing	2S-Adv Marg.	Adv3D Marg.	SSVM
Campus/ Stadtmitte	0.421 (0.07)	**0.453** (0.08)	0.424 (0.07)
Stadtmitte/ Campus	0.452 (0.10)	0.478 (0.11)	0.470 (0.09)
Bahnhof/ Sunnyday	0.552 (0.06)	**0.578** (0.05)	0.568 (0.05)
Pedcross2/ Sunnyday	0.535 (0.08)	**0.563** (0.08)	0.545 (0.10)
Sunnyday/ Bahnhof	0.541 (0.15)	**0.583** (0.17)	0.570 (0.18)
Pedcross2/ Bahnhof	0.565 (0.10)	**0.597** (0.13)	0.589 (0.16)
Bahnhof/ Pedcross2	0.492 (0.11)	**0.523** (0.11)	0.511 (0.13)
Sunnyday/ Pedcross2	0.499 (0.14)	**0.537** (0.12)	0.522 (0.14)

adversarial bipartite matching (2S-AdvMarg) proposed by [Fathony et al., 2018]. For the SSVM, we also use it to predict the best marginal distribution. We use `minConf` [Schmidt, 2008] to perform the projected Quasi-Newton optimization. In the prediction part of SSVM 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 SSVM).

4.3 RESULTS

Table 3 and Table 4 provide the mean and the standard deviation of the average accuracy for synthetic and video tracking data. It is calculated by $1 - \text{loss}_{Hamming}$.

Table 5: Running time (in seconds) with 50 samples

Dataset	# Objects	Adv3DMarg.	SSVM
Campus	12	21.34	11.72
Stadtmitte	16	54.73	18.79
Sunnyday	18	73.22	22.69
Pedcross2	30	357.70	146.12
Bahnhof	34	563.50	173.41

We can see that both 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. To compare the accuracy with SSVM, we use bold font to show the cases where Adv3DMarg outperformed with statistical significance. We can see that we have better results on all six pairs of the ETH datasets and still somewhat better than SSVM on the TUD datasets. For the synthetic data, the accuracy decreases as the number of objects increases, which is reasonable as the problem becomes harder.

To compare the running time, we list the time used on the video tracking dataset in Table 5. 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. 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 [Fathony et al., 2018], 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.

5 CONCLUSIONS & FUTURE WORK

In this paper, we use adversarial learning to formulate the 3D matching learning problem. We explore a way that avoids directly solving the 3D matching problem and can efficiently train on both synthetic and real datasets. Results of the average accuracy clearly show the improvement comparing with two stage bipartite matching approach. It also has better results over SSVM.

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.

Acknowledgements

The research in this paper is supported in part by NSF CAREER Award #1652530.

References

[Asif et al., 2015] Asif, K., Xing, W., Behpour, S., and Ziebart, B. D. (2015). Adversarial cost-sensitive classification. In *UAI*, pages 92–101.

[Behpour et al., 2018] Behpour, S., Xing, W., and Ziebart, B. D. (2018). ARC: Adversarial robust cuts for semi-supervised and multi-label classification. In *AAAI*, pages 2704–2711.

[Birkhoff, 1946] Birkhoff, G. (1946). Three observations on linear algebra. *Univ. Nac. Tacuman, Rev. Ser. A*, 5:147–151.

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

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

[Chari et al., 2015] Chari, V., Lacoste-Julien, S., Laptev, I., and Sivic, J. (2015). On pairwise costs for network flow multi-object tracking. In *CVPR*, pages 5537–5545.

[Cui et al., 2014] Cui, L.-B., Li, W., and Ng, M. K. (2014). Birkhoff–von neumann theorem for multi-stochastic tensors. *SIAM Journal on Matrix Analysis and Applications*, 35(3):956–973.

[Dalvi et al., 2004] Dalvi, N., Domingos, P., Sanghai, S., Verma, D., et al. (2004). Adversarial classification. In *KDD*, pages 99–108. ACM.

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

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

[Duchi et al., 2008] Duchi, J., Shalev-Shwartz, S., Singer, Y., and Chandra, T. (2008). Efficient projections onto the l 1-ball for learning in high dimensions. In *ICML*, pages 272–279. ACM.

[Fathony et al., 2017] Fathony, R., Bashiri, M. A., and Ziebart, B. (2017). Adversarial surrogate losses for ordinal regression. In *NeurIPS*, pages 563–573.

[Fathony et al., 2018] Fathony, R., Behpour, S., Zhang, X., and Ziebart, B. (2018). Efficient and consistent adversarial bipartite matching. In *ICML*, pages 1456–1465.

[Fathony et al., 2016] Fathony, R., Liu, A., Asif, K., and Ziebart, B. (2016). Adversarial multiclass classification: A risk minimization perspective. In *NeurIPS*, pages 559–567.

[Fisher, 1922] Fisher, R. A. (1922). 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.

[Goodfellow et al., 2014] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *NeurIPS*, pages 2672–2680.

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

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

[Joachims, 2005] Joachims, T. (2005). A support vector method for multivariate performance measures. In *ICML*, pages 377–384.

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

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

[Keuper et al., 2016] Keuper, M., Tang, S., Zhongjie, Y., Andres, B., Brox, T., and Schiele, B. (2016). A multicut formulation for joint segmentation and tracking of multiple objects. *arXiv preprint arXiv:1607.06317*.

[Kim et al., 2012] Kim, S., Kwak, S., Feyereisl, J., and Han, B. (2012). Online multi-target tracking by large margin structured learning. In *ACCV*, pages 98–111. Springer.

[Kuhn, 1955] Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics*, 2(1-2):83–97.

[Kulesza and Pereira, 2008] Kulesza, A. and Pereira, F. (2008). Structured learning with approximate inference. In *NeurIPS*, pages 785–792.

[Lafferty et al., 2001] Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289.

[Leal-Taixé et al., 2015] Leal-Taixé, L., Milan, A., Reid, I., Roth, S., and Schindler, K. (2015). Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv preprint arXiv:1504.01942*.

[Li et al., 2016] Li, J., Asif, K., Wang, H., Ziebart, B. D., and Berger-Wolf, T. Y. (2016). Adversarial sequence tagging. In *IJCAI*.

[Liu and Ziebart, 2014] Liu, A. and Ziebart, B. D. (2014). Robust classification under sample selection bias. In *NeurIPS*, pages 37–45.

[Liu and Han, 2015] Liu, L. and Han, Z. (2015). Multi-block admm for big data optimization in smart grid. In *ICNC*, pages 556–561. IEEE.

[Liu, 2007] Liu, Y. (2007). Fisher consistency of multicategory support vector machines. In *AISTATS*, pages 291–298.

[McMahan et al., 2003] McMahan, H. B., Gordon, G. J., and Blum, A. (2003). Planning in the presence of cost functions controlled by an adversary. In *ICML*, pages 536–543.

[Petterson et al., 2009] Petterson, J., Yu, J., McAuley, J. J., and Caetano, T. S. (2009). Exponential family graph matching and ranking. In *NeurIPS*, pages 1455–1463.

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

[Schmidt et al., 2009] Schmidt, M., Berg, E., Friedlander, M., and Murphy, K. (2009). Optimizing costly functions with simple constraints: A limited-memory projected quasi-Newton algorithm. In *AISTATS*, pages 456–463.

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

[Tang et al., 2015] Tang, S., Andres, B., Andriluka, M., and Schiele, B. (2015). Subgraph decomposition for multi-target tracking. In *CVPR*, pages 5033–5041.

[Tang et al., 2017] Tang, S., Andriluka, M., Andres, B., and Schiele, B. (2017). Multiple people tracking by lifted multicut and person re-identification. In *CVPR*, pages 3539–3548.

[Taskar et al., 2004] Taskar, B., Chatalbashev, V., and Koller, D. (2004). Learning associative markov networks. In *ICML*, page 102. ACM.

[Taskar et al., 2005] Taskar, B., Chatalbashev, V., Koller, D., and Guestrin, C. (2005). Learning structured prediction models: A large margin approach. In *ICML*, pages 896–903. ACM.

[Tewari and Bartlett, 2007] Tewari, A. and Bartlett, P. (2007). On the consistency of multiclass classification methods. *JMLR*, 8:1007–1025.

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

[Tsochantaridis et al., 2004] Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. (2004). Support vector machine learning for interdependent and structured output spaces. In *ICML*, page 104. ACM.

[Tsochantaridis et al., 2005] Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *JMLR*, 6(Sep):1453–1484.

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

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

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

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

[Zhang et al., 2008] Zhang, L., Li, Y., and Nevatia, R. (2008). Global data association for multi-object tracking using network flows. In *CVPR*, pages 1–8. IEEE.

Appendices

A DERIVING MARGINAL FORMULA FROM ORIGINAL PROBLEM

In this section we will show in details that the marginal form object function in Eq.7 is the same as Eq.5.

For a fixed x. 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:

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

B ADMM UPDATING FORMULA FOR EACH OF THE VARIABLES

Here we give the details about how we get the updating formula for $\mathbf{P}$. The key point is that modifying the object function by adding a constant or multiply a positive constant does not impact the process of finding the argmin. It is not hard to use the same method to get the updating formula of other variables.

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

C THE FEATURES USED IN THE EXPERIMENTS

Here we give more details about the features we used in the experiments.

C.1 Synthetic Data

The synthetic data contain these features:

1. A copy of the raw data : $X_{i,1}$,$X_{\pi_{1,i},2}$ and $X_{\pi_{2,i},3}$
2. summation of each pair of raw data in the given tuple, 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 raw data : $|X_{i,1}-X_{\pi_{1,i},2}|$, $|X_{i,1}-X_{\pi_{2,i},3}|$ and $|X_{\pi_{1,i},2}-X_{\pi_{2,i},3}|$
4. maximal and minimal L2 norm value of the raw data and 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 $|X_{\pi_{1,i},2}+X_{\pi_{2,i},3}-X_{i,1}|$

C.2 Video Tracking Data

The video tracking data contain these features:

1. Intersection over union (IoU) overlap ratio between bounding boxes. For pairs, IoU is area$(\mathrm{BB}_i^{t_1} \cap \mathrm{BB}_j^{t_2})/$ area$(\mathrm{BB}_i^{t_1} \cup \mathrm{BB}_j^{t_2})$, where BB_i^t denotes the bounding box of object i at time frame t. For triples, it is area$(\mathrm{BB}_i^{t_1} \cap \mathrm{BB}_j^{t_2} \cap \mathrm{BB}_k^{t_3})/$ area$(\mathrm{BB}_i^{t_1} \cup \mathrm{BB}_j^{t_2} \cup \mathrm{BB}_k^{t_3})$

2. Euclidean distance between object centers. For triples it will be the average distance between the nodes.

3. 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.

4. 21 local binary pattern (LBP) features from similar Bhattacharyaa distances and bounding box blocks. For triples we also use average value of the pairwise distance.

5. Optical flow (motion) between bounding boxes. For triples we also use average value of the pairwise distance of the affine transformation parameters.

6. Four indicator variables (for *entering*, *leaving*, *hiding in the middle* and *staying invisible).*

D ROBUSTNESS OF THE MODELS UNDER GAUSSIAN NOISE

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 3 that depict the case when $n = 5$, we can see that Adv3DMarg can slightly better keeping a high performance than SSVM.

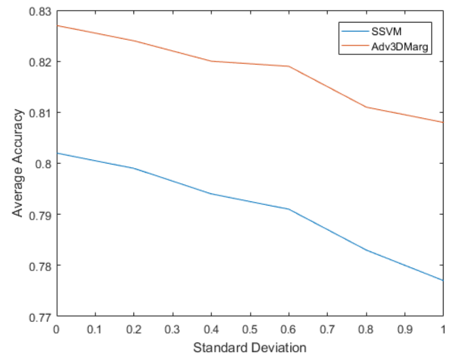

Figure 3: $n = 4$ Average accuracy when Gaussian noise added in $\{141, 233, 312, 424\ \}$