End-to-End Learning Deep CRF Models for Multi-Object Tracking Deep CRF Models

Jun Xiang, Guohan Xu, Chao Ma, and Jianhua Hou

Abstract—By bundling multiple complex sub-problems into a unified framework, end-to-end deep learning frameworks reduce the need for hand engineering or tuning of parameters for each component, and optimize different modules jointly to ensure the generalization of the whole deep architecture. Despite tremendous success in numerous computer vision tasks, end-to-end learnings for multi-object tracking (MOT), especially for the assignment problem in data association, have been surprisingly less investigated mainly due to the lack of available training data. Furthermore, it is challenging to discriminate target objects under mutual occlusions or to reduce identity switches in crowded scenes. To tackle these challenges, this paper proposes learning deep conditional random field (CRF) networks, aiming to model the assignment costs as unary potentials and the long-term dependencies among detection results as pairwise potentials. Specifically, we use a bidirectional long short-term memory (LSTM) network to encode the long-term dependencies. We pose the CRF inference as a recurrent neural network learning process using the standard gradient descent algorithm, where unary and pairwise potentials are jointly optimized in an end-to-end manner. Extensive experiments are conducted on the challenging MOT datasets including MOT15, MOT16 and MOT17, and the results show that the proposed algorithm performs favorably against the state-of-the-art methods.

Index Terms—Multi-object tracking, end-to-end deep learning, conditional random field, data association.

I. INTRODUCTION

MOST current multi-object tracking (MOT) approaches follow tracking-by-detection paradigm. Given detection responses by a pre-trained detector, the task of MOT is cast as a data association problem that consists of an affinity model for estimating the assignment cost between detections and tracklets, followed by an optimization strategy to determine which of the targets should be linked considering their affinity measurements [1].

With the recent rise of deep learning, end-to-end learning frameworks have been immensely successful in numerous sub-fields in vision community such as image recognition [2], [3], object detection [4]–[6], semantic segmentation [7], [8], single object tracking (SOT) [9], [10] etc. By bundling multiple complex sub-problems into a monolithic solution, this learning strategy shows its excellence with reducing the need for hand engineering or tuning of parameters for each component, and optimizing different modules jointly to ensure the generalization of the whole deep architecture. However, there are surprisingly few works [11], [12] that investigate multi-object tracking, especially the optimization for assignment problem, in an end-to-end manner [13], [14]. As identified by [11], the main reason for this is the lack of available training data to deal with a huge number of parameters and guarantee the generalizability of the trained deep architecture. Existing deep MOT algorithms put more emphasis on learning a favorable representation of target objects to build various affinity models [15]–[19]. In these approaches, traditional optimization strategies like bipartite graph matching [20] or linear assignment (LAP) [21] is employed, rendering the assignment problem beyond the range of deep learning.

Despite noticeable progress, a challenge still exists in most deep MOT methods on how to distinguish target objects in crowded scenes, where occlusions, noisy detections and large appearance variations often occur, due to the lack of the efficient component for modeling long-term dependencies among detection results over time. In MOT, the dependencies among targets mean the structural correlation of targets in the spatiotemporal domain, which are usually subject to a number of physical constraints. For instance, a target object cannot be at two places at one time, and two target objects cannot occupy the same space at the same time. Taking these constraints into consideration in data association helps to improve MOT performance. As shown in Fig. 1, person in the purple bounding boxes is partially occluded by person in the green bounding boxes. Without considering the dependencies among multiple targets, the unary similarity for purple marked person is only 0.45, which would fail to maintain a consistent identity at frame 67. In contrast, by modeling long-term dependencies between the two persons, the pairwise similarity of person purple marked is guided by person green marked (with higher unary similarity 0.64) and increases to 0.61, ensuring this target to be reliably tracked.

To address the above challenges, in this work we propose an end-to-end learning CRF for multi-object tracking within a unified deep architecture. The proposed method mainly consists of two steps: (i) designing task-specific networks to implicitly learn unary and pairwise potentials respectively;
(ii) conducting CRF inference as a recurrent neural network learning process using the standard gradient descent algorithm. Furthermore, the unary and pairwise components are pre-trained separately from scratch, and then fine-tuned along with the RNN inference component during backward pass. Compared with the monolithic end-to-end learning, one advantage of our two-step training strategy (i.e., pre-training components individually and then fine-tuning) lies in that relatively large amounts of training data could be exploited to learn each module for a specific task.

Our work partially resembles the recent CRF-RNN framework for semantic segmentation [7], [22], where the inference of CRF model is embedded into a recurrent neural network as well. However, the noticeable difference lies in that CRF-RNN is learned for determined pixel labeling rather than for graphical models with varying nodes (i.e., targets are varying from frame to frame in MOT). We model the unary terms by a matching network to compute the assignment cost. To fully exploit dependencies between detection results (i.e., CRF nodes), we pay more attention to difficult node pairs and formulate the labeling of these pairs as a joint probability matching problem. To be specific, we use a bidirectional LSTM to implicitly learn the pairwise terms. Finally, we pose the overall CRF inference as a recurrent neural network learning process using the standard gradient descent algorithm, where unary and pairwise potentials are jointly optimized in an end-to-end manner.

In summary, the main contributions of this work are as follows:

- We make the first attempt to learn deep CRF models for multi-object tracking in an end-to-end manner, where CRF potential learning and inference are integrated within a deep architecture.
- We propose a bidirectional LSTM to learn pairwise potentials that encode the long-term dependencies between CRF nodes.
- Our method is evaluated on the MOT15, MOT16 and MOT17 datasets, and achieves highly competitive performance over the state of the arts on the three benchmarks.

The proposed framework has been implemented in TensorFlow, and all source code will be made publicly available to facilitate further research.

II. RELATED WORK

A large amount of literature has developed on multi-object tracking. We refer to [13], [14] for a general review of this research direction. We also refer to [23] where Ciaparrone et al. provide the first comprehensive survey on the use of deep learning in MOT. In the tracking-by-detection framework, multi-object tracking consists of two critical components: a matching metric to compute assignment costs and a data association algorithm. Below, we briefly review the related work with the use of CRF and deep learning.

A. MOT With CRF

Before the prevalence of deep learning, CRFs have been used extensively for solving the assignment task in multi-object tracking [24]–[29], usually posed as an energy minimization problem. As a classical graphical model, CRFs relax the independent assumption among detections, aiming at using the long-term dependence among targets to better distinguish targets in crowded scenes. Yang et al. [26] adopt a CRF model and consider both tracklet affinities and dependencies among them, which are represented by unary term costs and pairwise term costs respectively. The approach in [24] further considers distinguishing difficult pairs of targets. While unary potentials are based on motion and appearance models for discriminating all targets, pairwise ones are designed for differentiating spatially close tracklets. Heil et al. [25], [27] perform association at the detection level. To exploit long-term connectivity between pairs of detections, the CRF is formulated in terms of similarity/dissimilarity pairwise factors, and additional higher order potentials are defined in terms of label costs. In [30], the energy term of CRF is augmented with a continuous component to jointly solve the discrete data association and continuous trajectory estimation problems.

However, MOT typically involves varying number of targets, which makes CRF inference intractable. As proved in [31], the energy function in MOT that only contains unary and pairwise terms does not follow the sub-modularity or regularity principle, and hence cannot be solved using standard optimization techniques like graph cuts. Existing approaches built on CRFs have to resort to heuristic [24] or iterative approximate algorithms [25], [27]. Moreover, these approaches assume that multiple targets are subject to Gaussian distribution [24], [25] when formulating the CRF potentials, and the Gaussian pairwise provides a smoothing term that encourages assigning similar labels to targets with similar properties. This is in contrast to the fact that there are both consistent and repellent dependencies between moving targets. Unlike existing approaches using heuristic or iterative approximate algorithms, we propose to learn deep CRF models as recurrent neural networks in an end-to-end manner.

Note that our method and approaches in [32], [33] are relevant to some extent since all of them could be categorized as multi-object tracking based on graph models. They share the similarity that the assignment hypothesis is finally cast as an energy minimization problem and solved using iterative approximate algorithms. However, ours differs from [32], [33] significantly. The authors in [32], [33] formulate MOT as a
graph partitioning problem and solve it via linear programming (LP), whereas we use a CRF framework and solve it by quadratic programming (QP). Furthermore, while the optimizations are performed beyond the range of neural networks in [32], [33], we pose the overall CRF inference (i.e., optimization) as a RNN learning process using the standard gradient descent algorithm.

B. MOT With Deep Learning

Recently, deep learning has been widely applied to multi-object tracking [15]–[19], [32], [34]–[36]. The trend in this line is to learn deep representations [16], [17], [19], and then employ traditional assignment strategies such as bipartite matching [16], [19], or linear assignment [17] for optimization. Leal-Taix et al. [15] propose a Siamese CNN to estimate the similarity between targets and detections. Tang et al. [32] go a step further by treating MOT as the person Re-ID problem and develop a Siamese ID-Net to compute association costs between detections. Sadeghian et al. [16] exploit CNN and LSTM to build the affinity measures based on appearance, motion and interaction cues. In [37], a deep affinity network (DAN) is proposed that jointly models object appearances and their affinities between different frames in an end-to-end fashion, and objects in the current frame are associated to the objects in multiple previous frames using the Hungarian algorithm. Deep metric learning is also proposed for representation learning so that the constructed affinity models are robust to appearance variations [17], [19], [35]. Zhou et al. [38] have recently proposed a deep continuous CRF model (DCCRF). While unary terms are modeled by a CNN to predict object displacements across time, the asymmetric pairwise terms are leveraged to address inter-object interactions for regularizing the final displacement. The DCCRF is learned end-to-end to compute affinity scores, but the association is performed using Hungarian algorithm.

While most of the approaches have exploited the representational power of deep neural networks (DNNs) for feature extraction and affinity computation, only very few approaches use deep learning to directly guide the assignment problem in data association [23]. Several works have taken step towards fully end-to-end learning for multi-target tracking.

Ondruska and Posner [39] are among the first who propose to replace the multi-stage pipeline in MOT with a single end-to-end learning framework, by introducing deep recurrent neural networks (RNN) to the task of state estimation. Although this work shows its efficacy, the limitation lies in the application to simulated data with near-perfect sensor measurements. Instead of pursuing the monolithic end-to-end approach as [39], Milan et al. [11] propose a full end-to-end learning approach by representing the state estimation and data association problems separately. They design a RNN based sub-network for Bayesian filtering as well as track management, and a Long Short-Term Memory (LSTM) based sub-network for data association. Taking as input the pairwise-distance matrix, the LSTMs output the probability matrix indicating target-to-measurement assignments, which is passed to RNN for state updating. Despite not using any visual features, their approach achieves reasonable performance relative to other similar systems. Schulter et al. [40] formulate the data association as a network flow problem solved by finding minimum cost network flow. They propose to learn parameterized cost functions with a deep neural network, which handles both affinity learning and data association in an end-to-end fashion. Shen et al. [41] extend the work in [40]. Instead of using detections as in [40], tracklets are introduced into the bi-level optimization framework to handle long time occlusions. They use deep metric learning to extract the appearance embedding for each target candidate, and the feature learning and data association are unified by a bi-level optimization formulation in end-to-end fashion. More recently, Chu and Ling [42] present a deep architecture, named FAMNet, where feature extraction, affinity metric and multi-dimensional assignment are learned jointly in an end-to-end fashion. Both appearance and motion clues are fused into the affinity model for more robust affinity metric, and the assignment is optimized following the rank-1 tensor approximation framework [43] by a power iteration with L1 normalization.

Different from the above works, we formulate multi-object tracking within a deep CRF framework. We pose the assignment optimization as a recurrent neural network learning process, where unary and pairwise potentials are jointly optimized in an end-to-end manner. Furthermore, a bidirectional LSTM network is utilized to encode the long-term dependencies among targets.

III. PROPOSED METHOD

In this section, we first describe how the data association based MOT is mapped into a CRF labeling problem. The designs of unary and pairwise potentials are then introduced. We present RNN based inference strategy and finally the overview of the entire structure.

A. Problem Formulation

Similar to [44], we treat the task of MOT as a CRF labeling problem. The flowchart of our approach is shown in Fig. 2. We use noisy detections as a starting point, then a two-threshold strategy [45] is employed to connect detection responses into short but reliable tracklets. A tracklet \( T_i = \{d_{i}^{0}, \ldots, d_{i}^{T}\} \) represents a set of detection responses in consecutive frames, starting at frame \( t_i^0 \) and ending at \( t_i^T \). Let \( d_{i}^{j} = \{p_{i}^{j}, s_{i}^{j}\} \) denotes a detection of target \( T_i \) at time step \( t \), with position \( p_{i}^{j} \) and size \( s_{i}^{j} \).

A graph \( G = (V, E) \) for a CRF is created over the set of tracklets \( T = \{T_i\} \), where \( V \) and \( E \) denote the set of nodes and edges, respectively. Tracklet \( T_i \) is linkable to \( T_j \) if the gap between the end of \( T_i \) and the beginning of \( T_j \) satisfies \( 0 < t_j^0 - t_i^T < T_{thr} \), where \( T_{thr} \) is a threshold for maximum gap between any linkable tracklet pair. A linkable pair of tracklets \( (T_i^1 \rightarrow T_i^2) \) forms the graph node \( v_i = (T_i^1 \rightarrow T_i^2), i = 1, 2, \ldots, |V| \). An edge \( e_{ij} = (v_{ij}, v_{ij}) \in E \) represents a correlation between a pair of nodes. Furthermore, each node is associated with a binary label variable \( x_i \in L = \{0, 1\} \), where \( x_i = 1 \) indicates the two tracklets in node \( v_i \) should be associated and \( x_i = 0 \) means the opposite.
Then \( X = [x_1, x_2, \cdots, x_{|V|}] \), a variable realization over all the nodes, corresponds to an association hypothesis of the set of tracklets \( T \). The pair of \((X, T)\) is modeled as a CRF characterized by the Gibbs distribution, formulated as

\[
P(X = \mathbf{x}|T) = \frac{1}{Z(T)} \exp(-E(\mathbf{x}|T))
\]

where \( E(\mathbf{x}|T) \) denotes the Gibbs energy function with respect to the labeling \( \mathbf{x} \in \mathcal{L}^{|V|} \), \( Z(T) \) is the partition function. For simplicity of notation, the conditioning on \( T \) will from now on be dropped. And the normalization term \( Z(T) \) in Eq. 1 can be omitted when solving for the maximum-probability labeling \( \mathbf{x}^* \) for a particular set of observations \( T \). In this paper, energy function is restricted containing a set of unary and pairwise terms and formulated as:

\[
E(\mathbf{x}) = \sum_{i \in V} \phi_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \phi_{ij}(x_i, x_j)
\]

where \( \phi_i : \mathcal{L} \rightarrow \mathbb{R} \) and \( \phi_{ij} : \mathcal{L} \times \mathcal{L} \rightarrow \mathbb{R} \) are the unary and pairwise potentials, respectively. As is common with CRFs, our tracking problem is transformed into an energy minimization problem.

B. Unary Potentials

The unary potential \( \phi_i(x_i) \) specifies the energy cost of assigning label \( x_i \) to node \( v_i \). In this paper we obtain our unary from a Siamese CNN with shared parameters. Roughly speaking, the Siamese Net outputs a probability of each linkable tracklet pair containing each label (i.e., belonging to the same target or not). Denoting the probability for node \( v_i \) and label \( x_i \) as \( z_{i|x_i} \), the unary potential is

\[
\phi_i(x_i) = -\omega_u \log(z_{i|x_i} + \epsilon)
\]

where \( \omega_u \) is a parameter controlling the impact of the unary potentials and \( \epsilon \) is introduced to avoid numerical problems for small values of \( z_{i|x_i} \).

C. Pairwise Potential

The pairwise potential \( \phi_{ij}(x_i, x_j) \) is defined as the energy cost of assigning label \( x_i \) to node \( v_i \) and \( x_j \) to \( v_j \) simultaneously. To address the challenge of association in crowded scenes, we propose a novel potential function built on difficult node pairs, and limit the number of edges by imposing pairwise terms on these pairs for efficiency.

1) Difficult Node Pairs: Difficult node pairs are defined as spatio-temporal head-close or tail-close tracklet pairs like in [44], which are usually caused by close tracklets crossing each other or by two targets interacting as a group. In this situation, we impose consistency constraints to restrict the labeling of difficult node pairs. Take Fig. 3(a) as a example. Since node \( v_i = (T_1 \rightarrow T_2) \) and \( v_j = (T_j \rightarrow T_2) \) are caused by close tracks crossing each other, a reasonable assumption is that if \( T_1 \) is associated to \( T_j \), then there is a high probability that \( T_j \) is associated to \( T_2 \) and vice versa, if \( T_1 \) is not associated to \( T_2 \), there is a high probability that \( T_j \) should not be associated to \( T_2 \).

Another kind of difficult node pairs come from the physical constraint that allows one hypothesis to be explained by at one target. We apply repellency constraints in this case. An illustration is shown in Fig. 3(b), where tracklet \( T_{11} = T_{j1} \) is shared by node \( v_i \) and \( v_j \). There exists a mutual exclusion to the labeling since two objects cannot occupy the same space at the same time. Specifically, labels of \( v_i \) and \( v_j \) cannot be set to “1” simultaneously due to time overlapping of the two nodes. It is worth noting that the labeling repellency is helpful to ensure that assignment problems meet the one-to-one relationship to prevent the same measurement to be assigned to multiple targets.

2) Potential Functions: So far, we have introduced labeling consistency/repellency constraints for difficult node pairs, which encode dependencies among targets and could be exploited for CRF inference. Different from existing MOT methods that are limited by hand-designed pairwise potentials, the key idea of our solution is to avoid the need to specify the explicit knowledge among difficult node pairs. Instead, we use highly expressive neural networks to learn their functions directly from the data.

In this paper, we formulate labeling of difficult node pairs as a joint matching problem. As illustrated in Fig. 4,
a bidirectional LSTM-based architecture is learned for this label assignment.

For a node pair \( \{v_i(T_1 \rightarrow T_2), v_j(T_1 \rightarrow T_2)\} \), we first build two tracklet sets \( A = \{T_1, T_1\} \) and \( B = \{T_2, T_2\} \) as our matching graph. The goal here is to find correspondences between tracklets in the two sets. Note that we do not impose constraints to maintain a one-to-one relationship between \( A \) and \( B \), i.e., points in \( A \) can be assigned to more than one points in \( B \).

The input feature \( F \) that connects visual and motion cues is first passed through an embedding layer and then fed into a two time step bidirectional LSTM [46]. The network sequentially outputs a joint probability distribution for the matching problem, i.e., one element from set \( A \) to each element in set \( B \). Here \( y \in \{0, 1\} \) is a binary decision variable representing an assignment between two tracklets.

The pairwise potential consists of appearance and motion cues between tracklets \( i \) and \( j \). Here \( y \in \{0, 1\} \) is a binary decision variable representing an assignment between two tracklets. Finally, we define the pairwise potential function for the difficult node pair \( \{v_i, v_j\} \) as

\[
\phi_{i,j}(x_i, x_j) = -\omega_p \log \left( z_{i;x_i} \cdot z_{j;x_j} + \varepsilon \right)
\]

where \( z_{i;x_i} = p(x_i|T_1, T_2) \) and \( z_{j;x_j} = p(x_j|T_1, T_2) \) represent the probabilities of joint matching state over \( v_i \) and \( v_j \). \( \omega_p \) is a parameter controlling the impact of potentials and \( \varepsilon \) is introduced to avoid numerical problems for small values of \( z_{i;x_i} \cdot z_{j;x_j} \).

3) Deep Feature \( F \): The deep feature \( F \) used to represent pairwise potentials consists of appearance and motion cues over node pairs as follows:

\[
F = [f_a(T_1), f_a(T_2), f_a(T_1), f_a(T_2), f_{m}^{i}(T_1, T_2), f_{m}^{j}(T_1, T_2), f_{m}^{i}(T_1, T_2), f_{m}^{j}(T_1, T_2), f_{\text{node}}(v_i, v_j)]
\]

In Eq. 5, the deep appearance feature \( f_a(T_1) \) is generated by inputting the most confident detection in \( T_1 \) to a Re-ID network (see Section IV.B.2 for details). \( f_{m}^{i}(T_k, T_m) \) means the motion feature between tracklet \( T_k \) and \( T_m \), defined by the distance between estimations of positions of two tracklets using a linear motion model and the real positions as in Kuo et al. [47] and Yang et al. [24]. As shown in Fig. 5(a), motion vector of \( T_k \) and \( T_m \) is defined as \( f_{m}^{i}(T_k, T_m) = [\Delta p_1, \Delta p_2] \), where distances are computed as \( \Delta p_1 = p_1^t_k + v^i_k \cdot (t_1^k - t_2^k) \) and \( \Delta p_2 = p_2^t_m - v^i_m \cdot (t_1^m - t_2^m) - p_2^t_m \). Here \( v^i_k \) stands for the velocity of tracklet \( T_k \) at time step \( t \). Finally, \( f_{\text{node}}(v_i, v_j) \) is a distance function for a difficult node pair. As shown in Fig. 5(b), considering a node pair \( \{v_i(T_1 \rightarrow T_2), v_j(T_1 \rightarrow T_2)\} \) and \( T_1 \) and \( T_2 \) are a head-close tracklet pair. Let \( t_x = \min(t_{1x}^{i}, t_{1x}^{j}) \), we estimate positions of both targets at frame \( t_x \), shown by black points in Fig. 5(b). The distance of the node pair is defined by the estimated relative distance and real position distance at frame \( t_x \), formulated as \( f_{\text{node}}(v_i, v_j) = [\Delta p_1, \Delta p_2] \), where relative distances \( \Delta p_1 = p_1^{i x} - p_1^{j x} - (t_{1x}^{i} - t_{1x}^{j} (t_{2x}^{j} - t_{2x}^{i})) \) and \( \Delta p_2 = p_2^{i x} - p_2^{j x} \).

D. Inference as RNN

One challenge in CRF inference is that a realization of assignment hypothesis takes values from a discrete set, e.g., \( \{0, 1\} \). However, existing deep learning methods are not designed for the discrete problem. To address this issue and use DNNs to produce optimal assignment hypothesis, we first take continuous relaxation of the original binary labelling and then pose it as an optimization problem like in [22]. Specifically, the original minimization for Eq. 2 is first transformed into an equivalent integer program problem by expanding label variable \( x_i \) into another new Boolean variables \( x_{i;\lambda}, \lambda \in \{0, 1\} \). \( x_{i;\lambda} \) represents an assignment of label \( \lambda \) to \( x_i \), which is equivalent to an assignment of Boolean labels 0 or 1 to each node \( x_{i;\lambda} \), and an assignment of label 1 to \( x_{i;\lambda} \) means that \( x_i \) receives label \( \lambda \). A constraint is introduced to ensure that only one label value is assigned to each node, i.e., \( \sum_{\lambda \in \mathcal{L}} x_{i;\lambda} = 1 \).

In this way, we rewrite the original energy minimization as the following binary integer program

\[
\min \sum_{i \in \mathcal{V}, \lambda \in \mathcal{L}} \phi_i(\lambda) x_{i;\lambda} + \sum_{i \in \mathcal{V}, \lambda \in \mathcal{L}} \sum_{\mu \in \mathcal{L}} \phi_{ij}(\lambda, \mu) x_{i;\lambda} x_{j;\mu} \\
\text{s.t. } x_{i;\lambda} \in \{0, 1\} \forall i \in \mathcal{V}, \lambda \in \mathcal{L} \\
\sum_{\lambda \in \mathcal{L}} x_{i;\lambda} = 1 \forall j \in \mathcal{V}
\]

As a next step, we relax the integer program by allowing real values on the unit interval \([0, 1]\) instead of Booleans only. Let \( q_{i;\lambda} \in [0, 1] \) denote the relaxed variables, and energy functions
can be represented as the following quadratic program

$$\min \sum_{i \in V, \lambda \in \mathcal{L}} \phi_i(\lambda) q_{i;\lambda} + \sum_{(i, j) \in \mathcal{E}} \phi_{ij}(\lambda, \mu) q_{i;\lambda} q_{j;\mu}$$

s.t. \( q_{i;\lambda} \in [0, 1] \quad \forall i \in V, \lambda \in \mathcal{L} \)

$$\sum_{\lambda \in \mathcal{L}} q_{i;\lambda} = 1 \quad \forall j \in V$$

(7)

By now, we can take the CRF inference in Eq. 7 as a gradient descent based minimization problem which can easily be formulated in a recurrent neural network manner, since all of the operations are differentiable with respect to \( q \). Below, we describe how to implement one iteration of gradient descent for our CRF inference with common operations of neural network layers.

1) One Iteration of CRF Inference: We use the standard gradient method to minimize the energy function in Eq. 7. The gradient of the objective function in Eq. 7, \( \nabla_q E \), has the following elements

$$\frac{\partial E}{\partial q_{i;\lambda}} = \phi_i(\lambda) + \sum_{(i, j) \in \mathcal{E}} \phi_{ij}(\lambda, \mu) q_{j;\mu}$$

(8)

We denote the contribution from pairwise term by

$$q'_{i;\lambda} = \sum_{(i, j) \in \mathcal{E}} \phi_{ij}(\lambda, \mu) q_{j;\mu}$$

(9)

and decompose \( q'_{i;\lambda} \) into the following two operations which can be described as CNN layers

$$q'_{i;\lambda}(\mu) = \sum_{(i, j) \in \mathcal{E}} \phi_{ij}(\lambda, \mu) q_{j;\mu}$$

(10)

$$q_{i;\lambda}^t(\mu) = \sum_{\mu \in \mathcal{L}} q'_{i;\lambda}(\mu)$$

(11)

In Eq. 10, \( q'_{i;\lambda}(\mu) \) is obtained by applying a filter on \( q_{j;\mu} \), which can be implemented through a standard convolution layer. Here the filter’s receptive field spans the whole nodes and coefficients of the filter are derived on the pairwise potential functions \( \phi_{ij}(\lambda, \mu) \). For \( q'_{i;\lambda} \) in Eq. 11, the operation step is to take a sum of the filter outputs for each class. Since each class label is considered individually, this can be viewed as usual convolution where the spatial receptive field of the filter is \( 1 \times 1 \), and the number of input or output channels is \( |\mathcal{L}| \). To get the complete gradient in Eq. 8, the output from the pairwise stage and the unary term \( \phi_i(\lambda) \) are combined using an element-wise summing operation. Then according to the gradient update principle, the new state of \( q \) can be computed by taking a step in the negative direction of gradient:

$$q_{i;\lambda}^{t+1} = q_i^t - \gamma \nabla_q E$$

(12)

where \( \gamma \) is the step size. Finally, we normalize our values to satisfy \( \sum_{i \in \mathcal{L}} q_{i;\lambda} = 1 \) and \( 0 \leq q_{i;\lambda} \leq 1 \) by applying a Softmax operation with no parameters.

Fig. 6. Data flow of one iteration of the gradient descent algorithm. Each rectangle or circle represents an operation that can be performed as the standard convolutional operations.

2) Integration in a Recurrent Neural Network: We have formulated one iteration of the CRF inference into a stack of common CNN operations. Now we can organize the entire CRF inference as RNN architecture, by unrolling the iterative gradient descent algorithm as a recurrent neural network and all the CRF’s parameters can be learned or fine-tuned end-to-end via back propagation. The data flow of one iteration of RNN inference is shown in Fig. 6. The RNN takes tracklets’ observations \( T \) for both unary and pairwise probability as input. Each iteration of RNN in the forward pass performs one step of gradient descent for CRF inference, denoted by \( q_{i;\lambda}^{t+1} = f(q_i^t, \omega, T) \), where \( \omega \) represents a set of parameters containing \( \omega_u, \omega_p, \gamma \) as well as filter weights for pairwise and unary terms. In the stage of initialization (i.e., at the first time step), \( q_0^t \) is set as the probability outputted by the unary part. At all other time steps, \( q_i^t \) is updated by \( q_{i;\lambda}^{t+1} \). The RNN outputs nothing but \( q_i^t \) in the last iteration that approximates a solution to Eq. 7. Finally, we binarize continuous probability scores \( q_i^t \) via thresholding to generate discrete labels. In practice, the repellency constraint works well. We have examined the testing results, and little or nothing occurs that a tracklet shared by two nodes is assigned to two different tracks. In the testing stage, tracklets are associated properly by binarizing continuous probability scores \( q_i^t \) with a threshold of 0.7.

E. Final Deep Structure Model

The entire structure of our framework is shown in Fig. 2. Given graph nodes, the first part of our model consists of a unary and pairwise architecture to produce potential values of nodes. The second part is a RNN model that performs CRF inference with gradient descent algorithm. Since all the pieces of the model are formulated into standard network operations, our final model can be trained in the manner of end-to-end using common deep learning strategies. To be specific, during forward pass, the input nodes are fed into CNNs and LSTM to compute unary and pairwise potentials. The output by CNN is then passed to RNN as initialization, i.e., CRF state \( q_0 \). After \( T \) iterations, the RNN finally outputs \( q_i^T \), the solution to Eq. 7. During backward pass, the error derivatives are first passed through the RNN, where the gradients w.r.t. the parameters and input accumulate through the \( T \) iterations. The gradients are back propagated to unary and pairwise components, and then update is performed for the CNN and LSTM simultaneously. In other word, all the CRF learning and inference are embedded within a unified neural network.

Note that due to limited MOT training data, we adopt a two-step training strategy (i.e., pre-training components
individually and then fine-tuning) to prevent the proposed network from overfitting. This strategy helps to speed up network training as well.

IV. IMPLEMENTATION DETAILS

This section describes hyperparameter choice and the details of the proposed method in both the training and testing stages. The entire deep architecture consists of three main components, where the unary and pairwise modules are trained separately from scratch, and then fine-tuned along with the learning of RNN inference component during backward pass. In addition, we employ a pre-trained Re-ID network to generate deep feature $F$. This network is fine-tuned using MOT datasets and not involved in the end-to-end training.

A. Hyperparameter Choices

The proposed CRF model has two learnable parameters. The first is the number of iterations $T$ in the gradient decent algorithm. In our experiments, we set the number of $T$ to 5 during the training stage to avoid the gradient vanishing problem. Fig. 7 shows the energy of Eq. 7 converges fast (after 5 iterations) on three test datasets.

The second parameter is $T_{thr}$, the maximal time interval of linkable tracklet pairs to build the graph. Intuitively, a small threshold $T_{thr}$ would be inefficient to deal with long-term missing detections, hence resulting in more tracking fragmentations. In experiments, we adopt two-level association to gradually link tracklets into final tracks. For all dataset, $T_{thr}$ is empirically set to 20. The lower the threshold is, the less the short tracklets are likely to be associated, leading to fewer ID switches but more fragments. In fact, missing detections, false detections and unassociated detections inevitably lead to fragmentations during the first round of association. To alleviate this problem, a second round association with $T_{thr} = 50$ is performed over the output of the first round to reconnect fragment tracks.

B. Architecture and Data Collection

We train our models using detection bounding boxes with associated IDs from ground-truth labels and randomly shift the center, width and height of the boxes. To avoid confusion, only detections whose visibility scores are larger than 0.5 are selected. We design an explicit occlusion reasoning model as in [29], [48] to compute the visibility for each target.

1) Unary Potentials: Our base CNN architecture is the VGG-16 model [3], denoted as ID-Net. We collect training set from two different datasets. We collect training images firstly from the 2DMOT2015 benchmark training set [49] and 5 sequences from the MOT16 benchmark training set [50]. We also collect person identity examples from the CUHK03 [51] and Market-1501 [52] datasets. For validation set, we use the MOT16-02 and MOT16-11 sequences from the MOT16 training set. In total, 2551 identities are used for training and 123 identities for validating. Specifically, we train VGG-16 from scratch with $Y = 2551$ unique identities. Training images are resized to $112 \times 224 \times 3$ and each image $I_i$ corresponds to a ground-truth identity label $l_i \in \{1, 2, \cdots, Y\}$. The network is trained with the softmax loss. A Siamese network is learned by fine-tuning the ID-Net with a binary softmax function. We set the batch size to 64, momentum to 0.9, dropout ratio for FC layers to 0.5. The learning rate is initially set to 0.01, and then decreased by a factor of 10 when accuracy in the validation set stopped improving. The maximum number of iterations is set to 40000.

2) Pairwise Potentials: To learn visual features, we directly employ a pre-trained person Re-Identification network [53], which is the state-of-the-art Re-ID network using a variant of triplet loss, and then fine-tune it on our MOT datasets. We collect triplet examples from the MOT15 benchmark training set and 5 sequences of the MOT16 benchmark training set. The MOT16-02, MOT16-09 sequences in the MOT-16 training set are used as testing sets. Overall 888 identities are used for training and 79 identities for validation. Triplet examples are generated as follows: for each batch of 100 instances, we select 5 persons and generate 20 instances for each person. In each triplet instance, the anchor and anchor positive are randomly selected from the same identity, and the negative one is also randomly selected but from the remaining identities. The triplet loss margin is set to -1. Here the final deep feature $F$ representing each node pair is a $4 \times 128 + 10 \times 2 = 532$ dimensional vector.

For the bidirectional LSTM, an FC layer is employed to map the entire input $F$ into a 300 dimensional embedding space. We use a two-layer hidden state vector of size 200 and a FC layer with dropout 0.5 to produce the $1 \times 2 \times 2$ distribution output at each step. During training, difficult node pairs are collected by randomly sampling tracklets from true target trajectories. Specifically, we collect true labels for those node pairs with spatial-temporal relationships as shown in Fig. 3(a) and Fig. 3(b). The labeling of these node pairs is then posed as a joint probability matching problem. Note that for a pair of repelling nodes, the joint ground-truth label could be (1,0), (0,1) or (0,0), except (1,1). Consequently, the one-to-one constraint is maintained in an implicit manner. We train a bidirectional LSTM from scratch via cross entropy loss, using 65000 samples which are divided almost equally between consistent and repellent pairs. The training data is divided into mini-batches where each batch has 32 samples. We normalize the position data to the range [-0.5, 0.5] with respect to image dimensions. We use the Adam algorithm [54] to minimize the cross-entropy loss. The learning rate is set to 0.0001 and the maximum number of iterations is set to 40000 for convergence.
In the testing stage, the LSTM predicts the joint distribution for matching state and computes the pairwise potentials.

3) CRF Inference: We generate tracklets by randomly sampling 6 sequences of the MOT16 dataset as training data and use the MOT16-02 and MOT16-09 sequence as the validation set. The learning parameters of the RNN are the impact coefficients \(w_u, w_p\), and learning step \(\gamma\). We fine-tune unary and pairwise potentials simultaneously. To handle the issue of varying number of targets in learning neural network architectures, we adopt the sliding window strategy to produce a fixed number of nodes, then shift the window forward to ensure that the new sliding window overlaps with the previous region. This allows tracklets to be linked over time. Considering the trade-off between accuracy and efficiency, the node number in sliding window is set to 200 during both training and testing stages, and there is 50% overlap (i.e., 100 nodes) between neighboring sliding windows. We use the Adam algorithm [54] to minimize the cross-entropy loss with 200 nodes per batch. The learning rate is set to 0.001. \(w_u, w_p\) are all initialized to 1, and the initial learning step \(\gamma\) is set to 0.5. The maximum number of iterations is set to 60000.

To obtain the final trajectories, a post-processing step is employed to improve the association results. First, short tracks fewer than 4 frames are removed, since these tracks are most likely to come from false alarms. Second, a linear motion model is employed for trajectory interpolation over gaps. Finally, bounding box regression (BBR) [55] is performed for better location.

V. EXPERIMENTAL RESULTS

In this section, we first describe the used evaluation metrics, and then present ablation studies on each component of our CRF framework. We evaluate the proposed method in comparison with state-of-the-art approaches on three MOT benchmark datasets. We also compare our method with its competitors that attempt to solve MOT in an end-to-end fashion. Finally, an analysis of the tracking failures is given.

A. Evaluation Metrics

We follow the standard MOT2D Benchmark challenge [49], [50] for evaluating multi-object tracking performance. These metrics include: Multiple Object Tracking Accuracy (MOTA \(\uparrow\)), The Ratio of Correctly Identified Detections (IDF1 \(\uparrow\)), Multiple Object Tracking Precision (MOTP \(\uparrow\)), Mostly Tracked targets (MT \(\uparrow\)), Mostly Lost targets (ML \(\downarrow\)), False Positives (FP \(\downarrow\)), False Negatives (FN \(\downarrow\)), Fragmentation (FM \(\downarrow\)), ID Switches (IDS \(\downarrow\)), and finally Processing speed (HZ \(\uparrow\)). The up arrow \(\uparrow\) denotes that the higher value is better and the down arrow \(\downarrow\) represents the opposite.

B. Ablation Studies

1) Different Components: Following the standard CRF based MOT framework, we investigate the contribution of different components in our method with detailed tracking metrics. The visualized results on the MOT16-02 and MOT16-09 datasets are also provided. To this end, we evaluate three alternative implementations of our algorithm with the same deep architectures as follows:

- Unary terms only (U). We use the Hungarian algorithm to find the globally optimal solution. In this case, we only consider appearance information of targets.
- CRF with an approximate solution in polynomial time (U+P). We use unary and pairwise terms but without global RNN inference. We replace the potentials in [24] with ours and conduct CRF inference with the heuristic algorithm [24], which indicates the efficiency of our learned potentials using the conventional optimization.
- The proposed method (CRF-RNN) equipped with all the components.

Table I shows that unary terms are effective for tracking, meaning that the appearance cue is important for MOT. The pairwise terms help to improve performance, especially in highly crowded scenes with clutter and occlusions. Comparing the U+P method with the U, MOTA and IDF1 are increased from 25.3% to 26.7% and 20.2% to 25.7% respectively. By using pairwise terms, spatial-temporal dependencies can be effectively modeled to distinguish close targets.

![Figure 8](image)

Fig. 8 visually compares the results of these two implementations. In the first row of Fig. 8(a), a fragment occurs at frame 283 when a little girl in white makes a sharp direction change which leads to failure of the appearance model. In the second row of Fig. 8(a), by incorporating pairwise terms, the U+P baseline is able to locate the non-linear path and track the girl correctly. In the first row of Fig. 8(b), an ID switch occurs between two close persons due to occlusions. However, by using pairwise terms, the U+P method can distinguish the targets, as shown in the second row of Fig. 8(b). Fig. 8(c) indicates that the pairwise model is also robust to illumination variations.

In contrast, the proposed CRF-RNN method outperforms both the U and U+P baselines in all metrics. This clearly shows that during CRF inference both unary and pairwise cues are complementary to contribute to tracking performance improvement. Fig. 9 also shows that our method is more robust to heavy occlusion, crowded scenes and illumination varying.

To assess the impact of the proposed end-to-end training, we also conduct an experiment by fixing the unary and pairwise parts while learning the RNN component (i.e., the CRF inference model) separately. In this case, the increment of MOTA is only 0.5% over the U+P baseline. On the other hand, our CRF-RNN surpasses the U+P with 2.1% MOTA (28.8% versus 26.7%). It is clear that end-to-end training is helpful to boost the tracking accuracy significantly.

Table II gives the results of a similar experiment conducted on the 2DMOT2015 validation set including TUD-Campus, ETH-SunCITY, ETH-Pedcross2 and KITTI-17. The trend of performance gains reported in Table II is consistent with that in Table I.

| Tracker | MOTA \(\uparrow\) | IDF1 \(\uparrow\) | MOTP \(\uparrow\) | MOT \(\uparrow\) | ML \(\downarrow\) | FP \(\downarrow\) | FN \(\downarrow\) | IDS \(\downarrow\) | HZ \(\uparrow\) |
|----------|----------------|---------------|----------------|----------------|-------------|-------------|-------------|-------------|-------------|
| U        | 25.3           | 20.2          | 34.9           | 11.4           | 50.8%       | 800         | 16294       | 339         |
| U+P      | 26.7           | 25.7          | 75.0           | 11.4           | 50.8%       | 323         | 16188       | 281         |
| CRF-RNN  | 28.8           | 31.4          | 75.1           | 11.4           | 44.1%       | 317         | 15943       | 173         |
Fig. 8. Qualitative comparison. The first row shows tracking results using unary terms only (U). The second row shows tracking results considering both unary and pairwise terms (U+P).

Fig. 9. Qualitative comparison. The first row shows tracking results of considering both unary and pairwise terms with the regular optimization algorithm (U+P). The second row shows the tracking results of our deep CRF method.

TABLE II
ABLATION STUDIES ON THE 2DMOT2015 VALIDATION SET

| Tracker | MOTA | IDF1 | MOTP | MT+ | ML+ | PP+ | PN+ | IDS  |
|---------|------|------|------|-----|-----|-----|-----|------|
| U       | 23.3 | 32.3 | 72.6 | 7.8%| 56% | 664 | 6143 | 199  |
| U+P     | 23.5 | 33.4 | 72.1 | 7.8%| 56% | 758 | 5944 | 124  |
| CRF-RNN | 28.4 | 41.4 | 72.0 | 7.8%| 56% | 498 | 5995 | 66   |

2) Analysis of the Potential Component: One key ingredient of the proposed method is the pairwise potentials built on difficult node pairs. Next, we design three experiments to provide an in-depth analysis regarding this component.

First, to validate the necessity of the LSTM, an ablation study is conducted by cutting the links between the LSTM modules and using two-layer FC (also with size 200 and dropout 0.5) to replace the LSTM. The pairwise module is illustrated in Fig. 10. In this situation, the module takes feature $F$ as input and produces the $2 \times 2 \times 2$ joint labeling distributions over node pairs. The experiment results are shown in Table III, where U+P is the same as Table I (using LSTM), and U+P (FC) means using two-layer FC instead of LSTM in the pairwise module. The test shows the performance gains brought by the LSTM. The method U+P outperforms U+P (FC) in terms of MOTA (19.6% versus 19%), IDF1 (20.7% versus 19.1%) and IDS (140 versus 180).

Second, 65000 difficult node pairs are picked to train the LSTM in this paper. What happens if we randomly sample node pairs? As described above, we define difficult node pairs over spatial-temporal head-close or tail-close tracklet pairs, which model the labeling correlations between a pair of nodes by consistent or repellent constraint. Generally, since the spatial-temporal relationships scarcely exist among the
Fig. 10. A ablation study regardin g the necessary of the LSTM by using two-layer FC to replace LSTM modules.

Fig. 11. The impact of the percentage of difficult node pairs on the labeling accuracy.

TABLE III
ANALYSIS OF THE NECESSITY OF THE LSTM ON MOT16-02 VALIDATION SET

| Tracker | MOTA | IDF1 | MOTP | MT | ML | FP | PN | IDS |
|---------|-------|------|------|----|----|----|----|-----|
| U(trip) | 19.6  | 20.7 | 75.4 | 7.4 | 66 | 373| 13821| 140 |
| U(trip)+P | 19.0 | 19.1 | 75.6 | 7.4 | 66 | 175| 14084| 180 |

majority of nodes (i.e., ordinary nodes), we only use possible nodes (about 20% in total) to construct the set of difficult node pairs. Pairs comprised by ordinary nodes encode little labeling correlation, and hardly contribute to learning dependencies between CRF nodes.

We conduct an additional experiment to validate the above claim. 40000 training samples and 5000 validation ones are collected from the MOT16 benchmark training set. Both difficult and ordinary node pairs are contained in the training set. In the testing phase, we collect 4957 difficult node pairs from MOT6-02 sequence. We evaluate the labeling accuracy over these testing samples by changing the percentage of difficult node pairs in training set. The results are shown in Fig. 11, which clearly indicates that increasing the number of difficult node pairs helps to improve the labeling accuracy. Compared with using ordinary node pairs merely, picking difficult node pairs as training set yields 8.9% performance gain (from 82.3% to 91.2%).

Finally, since two separate sets of visual features are used for unary and pairwise potentials, embedding multiple appearance, rather than P itself, might yields the performance gain for U+P in Table I.

TABLE IV
ABLATION STUDIES ON MOT16-02 VALIDATION SET (USING THE SAME APPEARANCE FEATURES)

| Tracker   | MOTA | IDF1 | MOTP | MT | ML | FP | PN | IDS |
|-----------|------|------|------|----|----|----|----|-----|
| U(trip)   | 18.4 | 18.8 | 75.4 | 7.4 | 66 | 274| 14083| 198 |
| U(trip)+P | 19.7 | 24.7 | 75.5 | 7.4 | 66 | 345| 13859| 131 |

To clarify this issue, we design a variant of unary component, termed as U(trip), by replacing the Siamese CNN with the Re-ID network used in [53]. An ablation experiment is then conducted by comparing the U(trip)+P method with the U(trip) only. The results are shown in Table IV. The U(trip)+P outperforms the U(trip) in terms of MOTA (19.7% versus 18.4%), IDF1 (24.7% versus 18.8%) and IDS (131 versus 198). It is evident that the performance gain comes from pairwise potentials rather than the model embedding, since both unary and pairwise models use the same appearance features here.

C. Comparison With the State of the Art

In this subsection, we evaluate our CRF-RNN method on the MOT15, MOT16 and MOT17 datasets. As adopted in most literature, we focus on the MOTA value as the main evaluation metric, which is a weighted combination of false negatives (FN), false positives (FP) and identity switches (IDS).

Table V presents the quantitative results on the MOT15 test set. Our method achieves the state-of-the-art performance in terms of MOTA (40%), IDF1(49.6%), and FN (25917). We obtain the second best record in terms of MT(23.0%) and ML(28.6%). The higher MT and lower ML suggest that our method is capable of recovering targets from occlusion or drifting. The comparing results on MOT16 benchmark are shown in Table VI. Similar to above discussion, our method outperforms other approaches on multiple metrics such as MOTA (50.3%), ML (35.7%) and FN (82746). The proposed method also ranks the 2nd with MT of 18.3% and the 3rd with IDF1 of 54.4%. We conduct experiment on MOT17 dataset and report the results in Table VII. While our MOTA (53.1%) and FN (234991) are the second best on the leader board, we obtain the state-of-the-art performance in terms of MT (24.2%) and ML (30.7%). Note that the approach in [57] has achieved so far the best MOTA on MOT15 and MOT16 benchmarks. These results are listed, but not compared due to later submission.

D. Comparison With Relevant Approaches

To further demonstrate its efficacy, the proposed method is compared with three approaches [11], [41], [42] that attempt to solve MOT in an end-to-end fashion. We also consider the algorithm in [38], where a deep continuous conditional random field (DCCRF) model trained end-to-end is proposed. The comparison results are shown in Table VIII.

Among them, RNNLSTM [11] is the first end-to-end learning approach for online MOT. Although the results on the

1Since our results come from the ArXiv paper [56] submitted on 29 Jul. 2019, we only consider officially peer-reviewed and published entries in the benchmarks before Aug. 2019.
Table V

| Tracker              | Mode  | MOTA | IDF1 | MOTOP | MT  | ML  | FP  | FN  | IDS  | HZ  |
|----------------------|-------|------|------|-------|-----|-----|-----|-----|------|-----|
| CRF-RNN(Ours)        | Batch | 40.0 | 49.6 | 71.9  | 23.0% | 28.6% | 10295 | 25917 | 658  | 3.2 |
| JointMC [58]         | Batch | 35.6 | 45.1 | 71.9  | 23.2% | 39.3% | 10580 | 28508 | 457  | 0.6 |
| Quad-CNN [19]        | Batch | 33.8 | 40.4 | 73.4  | 12.9% | 36.9% | 7898  | 32061 | 703  | 3.7 |
| NMT [28]             | Batch | 33.7 | 44.6 | 71.9  | 12.2% | 44.0% | 7762  | 32547 | 442  | 11.5|
| MHT-DAM [18]         | Batch | 32.4 | 45.3 | 71.8  | 16.0% | 43.8% | 9064  | 32060 | 435  | 0.7 |
| SiameseCNN [15]      | Batch | 32.0 | 34.3 | 71.2  | 8.5%  | 48.4% | 5160  | 37798 | 639  | 52.8|

Table VI

| Tracker              | Mode  | MOTA | IDF1 | MOTOP | MT  | ML  | FP  | FN  | IDS  | HZ  |
|----------------------|-------|------|------|-------|-----|-----|-----|-----|------|-----|
| KCF [59]             | Online | 38.9 | 44.5 | 70.6  | 16.6% | 31.5% | 7352  | 29501 | 720  | 0.3 |
| API-WDP [60]         | Online | 38.5 | 47.1 | 72.6  | 8.7%  | 37.4% | 4005  | 32303 | 586  | 6.7 |
| AMIR [16]            | Online | 37.6 | 46.0 | 71.7  | 15.8% | 26.8% | 7933  | 29397 | 1026 | 1.9 |
| RAR15pub [34]        | Online | 35.1 | 45.4 | 70.9  | 13.0% | 42.3% | 6771  | 32717 | 381  | 5.4 |
| CDA [61]             | Online | 32.8 | 38.8 | 70.7  | 9.7%  | 42.2% | 4983  | 35690 | 614  | 2.3 |
| Tracktor15 [57]      | Online | 44.1 | 46.7 | 75.0  | 18.0% | 26.2% | 6477  | 26577 | 1318 | 0.9 |

Table VII

| Tracker              | Mode  | MOTA | IDF1 | MOTOP | MT  | ML  | FP  | FN  | IDS  | HZ  |
|----------------------|-------|------|------|-------|-----|-----|-----|-----|------|-----|
| CRF-RNN(Ours)        | Batch | 50.3 | 54.4 | 74.8  | 18.3% | 35.7% | 7148  | 82746 | 702  | 1.5 |
| HCC [62]             | Batch | 49.3 | 50.7 | 79.0  | 17.8% | 39.9% | 5333  | 86795 | 391  | 0.8 |
| LMP [32]             | Batch | 48.8 | 51.3 | 79.0  | 18.2% | 40.1% | 6654  | 86245 | 481  | 0.5 |
| TL-MHT [63]          | Batch | 48.7 | 55.3 | 76.4  | 15.7% | 44.5% | 6632  | 86504 | 413  | 4.8 |
| ORCA [64]            | Batch | 48.2 | 48.6 | 77.5  | 12.9% | 41.1% | 5104  | 88586 | 821  | 2.8 |
| eHAF16 [65]          | Batch | 47.2 | 52.4 | 75.7  | 18.6% | 42.8% | 12586 | 83107 | 542  | 0.5 |
| NMT [28]             | Batch | 46.4 | 53.3 | 76.6  | 18.3% | 41.1% | 9753  | 87565 | 359  | 2.6 |
| Quad-CNN [19]        | Batch | 44.1 | 38.3 | 76.4  | 14.6% | 44.9% | 6388  | 94775 | 745  | 1.8 |
| MHT-DAM [18]         | Batch | 42.9 | 47.8 | 76.6  | 13.6% | 46.9% | 5668  | 97919 | 499  | 0.8 |

MOT15 test set do not quite reach top accuracy at the publishing time, the algorithm is two orders of magnitude faster (165 FPS) and shows the improvement potential by incorporating any kind of appearance features. AFN [41] can be viewed as an improved version of [40] which formulates MOT as a network flow problem solved via a linear programming (LP), whereas ours is a CRF framework which is equivalent to quadratic programming. FAMNet [42] presents an end-to-end model where feature extraction, affinity estimation and assignment are integrated into a deep architecture. Our method has better MOTA, IDF1, ML, FN and IDS on all the three MOT benchmarks compared to AFN [41] and FAMNet [42], which we believe could be attributed to the proposed deep CRF-RNN model as well as its capability to encode the long-term dependencies between targets.

DCCRF [38] is another relevant method proposed recently, where a deep end-to-end learned CRF model is utilized to compute affinity scores, and optimization is conducted by the traditional Hungarian algorithm. In contrast, we integrate CRF potential learning and inference into the proposed CRF-RNN model to allow for end-to-end learning. Our method outperforms DCCRF [38] by a relative large margin on most metrics, e.g., MOTA (40% versus 33.6% in MOT15, 50.3% versus 44.8% in MOT16), MT (23% versus 10.4% in MOT15,
TABLE VIII
THE RESULTS OF COMPARISON WITH RELEVANT APPROACHES ON THREE MOT DATASETS. THE BEST AND SECOND BEST RESULTS ARE HIGHLIGHTED BY BOLD AND RED FONTS

| Method          | Mode  | MOTA ↑ | IDF1 ↑ | MOTP ↑ | MT ↑ | ML ↓ | FP ↓ | FN ↓ | IDS ↓ | HZ ↑ |
|-----------------|-------|--------|--------|--------|------|------|------|------|-------|------|
| CRF-RNN(Ours)   | Batch | 40.0   | 49.6   | 71.9   | 23.0%| 28.6%| 10295| 25917| 658   | 3.2  |
| FAMNet [42]     | Online|        |        |        |      |      |      |      |       |      |
| APN [41]        | Batch | –      | –      | –      | –    | –    | –    | –    | –     | –    |
| DCCRF [38]      | Online| 33.6   | 39.1   | 70.9   | 10.4%| 37.6%| 5917 | 34002| 866   | 0.1  |
| RNN-LSTM [11]   | Online| 19     | 17.1   | 71     | 5.5% | 45.6%| 11578| 36706| 1490  | 165.2|
| 2DMOT2015       |       |        |        |        |      |      |      |      |       |      |
| CRF-RNN(Ours)   | Batch | 50.3   | 54.4   | 74.8   | 18.3%| 35.7%| 7148 | 82746| 702   | 1.5  |
| FAMNet [42]     | Online|        |        |        |      |      |      |      |       |      |
| APN [41]        | Batch | 49.0   | 48.2   | 78.0   | 19.1%| 35.7%| 9508 | 82506| 899   | 0.6  |
| DCCRF [38]      | Online| 44.8   | 39.7   | 75.6   | 14.1%| 42.3%| 5613 | 94125| 968   | 0.1  |
| RNN-LSTM [11]   | Online| –      | –      | –      | –    | –    | –    | –    | –     | –    |
| MOT16           |       |        |        |        |      |      |      |      |       |      |
| CRF-RNN(Ours)   | Batch | 53.1   | 53.7   | 76.1   | 24.2%| 30.7%| 27194| 234991| 2518  | 1.4  |
| FAMNet [42]     | Online| 52.0   | 48.7   | 76.5   | 19.1%| 33.4%| 14138| 253616| 3072  | –    |
| APN [41]        | Batch | 51.5   | 46.9   | 77.6   | 20.6%| 35.5%| 22391| 248420| 2593  | 1.8  |
| DCCRF [38]      | Online| –      | –      | –      | –    | –    | –    | –    | –     | –    |
| RNN-LSTM [11]   | Online| –      | –      | –      | –    | –    | –    | –    | –     | –    |

Fig. 12. An example of tracking failures.

18.3% versus 14.1% in MOT16). The distinct advantage over DCCRF indicates the potential to directly guide the assignment problem in MOT using deep CRF.

Note that all algorithms listed in Table VIII have been evaluated on only one or two MOT datasets except our CRF-RNN. Considering the best overall performance on all the three datasets compared to the existing relevant approaches, the proposed method has better robustness when faced with more complex scenarios.

E. Analysis of the Tracking Failure

Although the proposed CRF-RNN method works well in most cases, few failure cases occur in some unusual scenarios. An example is illustrated in Fig. 12, where a person in a white coat is occluded by a street lamp for a long time. As no detection is provided after frame 527 due to the occlusion, an ID switch happens from ID25 to ID177 when the person reappeared at frame 593.

The reasons behind it are twofold. First, the target has undergone a large appearance variation in a long time span. As our model is trained by detections whose visibility scores are larger than 0.5, the tracking is limited when faced with a heavily occluded target. In this situation, missed detections along with the varying appearance are likely to lead to a number of IDS. A potential means to address this classical challenge is to design occlusion reasoning model or part based model so that the occlusion relationship among targets could be explicitly inferred more effectively. Second, to encode the long-term dependencies among targets, one key ingredient of the paper is the potential function built on difficult node pairs. If there exists a spatially near target which construct a pair of difficult nodes with the concerned person, the consistency constraint could be helpful to alleviating the tracking failure. However, no such targets are available for the person in Fig. 12, and this issue lies beyond the scope of this paper.

Fig. 13 gives another typical failure example caused by inaccurate localizations from the detector. In the beginning, the target (person with a shoulder bag) is detected precisely, and our tracker works well. The detector subsequently suffers from drift as a man with a handbag is coming across the target. The tracker fails at frame 420 when more than half of the target is removed from the detection bounding box. This failure further leads to the issue that when re-detected in frame 421, the target is wrongly identified as a new comer by the tracker, since the appearance in the purple box largely differs from ID42 (the man with a handbag) at frame 420. Our future work is to introduce saliency region labelling to model the tracked target. This strategy helps maintain the consistency of the target in the foreground, thus suppressing background or other interfering objects.

VI. CONCLUSION

In this paper, we present a deep learning based CRF framework for multi-object tracking. To exploit the dependencies between detection results, we pay more attention to
handle difficult node pairs when modeling pairwise potentials. Specifically, we use bidirectional LSTMs to solve this joint probability matching problem. We pose the CRF inference as an RNN learning process using the standard gradient descent algorithm, where unary and pairwise potentials are jointly optimized in an end-to-end manner. Extensive experimental results on the challenging MOT datasets including MOT15, MOT16 and MOT17 demonstrate that the proposed deep CRF model is among the best performing techniques on the leader board for these challenges. Compared with the existing approaches that solve MOT in an end-to-end learning framework, our method also surpasses its competitors on a majority of the evaluation metrics.

REFERENCES
[1] C. Huang, Y. Li, and R. Nevatia, “Multiple target tracking by learning-based hierarchical association of detection responses,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 4, pp. 898–910, Apr. 2013.
[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Commun. ACM, vol. 60, no. 6, pp. 84–90, Jun. 2017.
[3] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014, arXiv:1409.1556. [Online]. Available: https://arxiv.org/abs/1409.1556
[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 779–788.
[5] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, Jun. 2017.
[6] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2961–2969.
[7] H. Liu et al., “Conditional random fields as recurrent neural networks,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 1529–1537.
[8] H. Liu et al., “An End-To-End network for panoptic segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 6127–6181.
[9] J. Valmadre, L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, “Fully-convolutional siamese networks for object tracking,” in Proc. Eur. Conf. Comput. Vis., 2016, pp. 850–865.
[10] J. Bertinetto, J. Valmadre, J. Henriques, A. Vedaldi, and P. H. S. Torr, “End-to-End representation learning for correlation filter based tracking,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2452–2461.
[11] A. Milan, S. H. Rezatofighi, A. R. Dick, K. Schindler, and I. D. Reid, “Online multi-target tracking using recurrent neural networks,” in Proc. 31st AAAI Conf. Artif. Intell., 2017, pp. 1–8.
[12] B. Beyer, S. Breuers, V. Kurin, and B. Leibe, “Towards a principled integration of multi-camera re-identification and tracking through optimal bayes filters,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jul. 2017, pp. 1444–1453.
[13] W. Luo et al., “Multiple object tracking: A literature review,” 2014, arXiv:1409.7616. [Online]. Available: https://arxiv.org/abs/1409.7618
[14] P. Emami, P. M. Pardalos, L. Elefteriadou, and S. Ranka, “Machine learning methods for solving assignment problems in multi-target tracking,” 2018, arXiv:1802.06897. [Online]. Available: https://arxiv.org/abs/1802.06897
[15] L. Leal-Taixe, C. Canton-Ferrer, and K. Schindler, “Learning by tracking: Siamese CNN for robust target association,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jun. 2016, pp. 418–425.
[16] A. Sadeghian, A. Alahi, and S. Savarese, “Tracking the untrackable: Learning to track multiple cues with long-term dependencies,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 300–311.
[17] B. Wang et al., “Joint learning of convolutional neural networks and temporally constrained metrics for tracklet association,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jun. 2016, pp. 386–393.
[18] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, “Multiple hypothesis tracking revisited,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 4696–4704.
[43] X. Shi, H. Ling, Y. Pang, W. Hu, P. Chu, and J. Xing, “Rank-1 tensor approximation for high-order association in multi-target tracking,” Int. J. Comput. Vis., vol. 127, no. 8, pp. 1063–1083, Jan. 2019.

[44] B. Yang and R. Nevatia, “An online learned CRF model for multi-target tracking,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2012, pp. 2034–2041.

[45] C. Huang, B. Wu, and R. Nevatia, “Robust object tracking by hierarchical association of detection responses,” in Proc. ECCV, 2008, pp. 788–801.

[46] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[47] C.-H. Kuo, C. Huang, and R. Nevatia, “Multi-target tracking by on-line learned discriminative appearance models,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., Jun. 2010, pp. 685–692.

[48] A. Milan, S. Roth, and K. Schindler, “Continuous energy minimization for multitarget tracking,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, no. 1, pp. 58–72, Jan. 2014.

[49] L. Leal-Taixe, A. Milan, I. Reid, S. Roth, and K. Schindler, “MOTChallenge 2015: Towards a benchmark for multi-target tracking,” 2015, arXiv:1504.01942. [Online]. Available: http://arxiv.org/abs/1504.01942

[50] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, “MOT16: A benchmark for multi-object tracking,” 2016, arXiv:1603.00831. [Online]. Available: http://arxiv.org/abs/1603.00831

[51] W. Li, R. Zhao, T. Xiao, and X. Wang, “DeepReID: Deep filter pairing neural network for person re-identification,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 152–159.

[52] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, “Scalable person re-identification: A benchmark,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 1116–1124.

[53] A. Hermans, L. Beyer, and B. Leibe, “In defense of the triplet loss for person re-identification,” 2017, arXiv:1703.07737. [Online]. Available: https://arxiv.org/abs/1703.07737

[54] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv:1412.6980. [Online]. Available: https://arxiv.org/abs/1412.6980

[55] R. Girshick, “Fast R-CNN,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 1440–1448.

[56] J. Xiang, M. Chao, G. Xu, and J. Hou, “End-to-End learning deep CRF models for multi-object tracking,” 2019, arXiv:1907.12176. [Online]. Available: http://arxiv.org/abs/1907.12176

[57] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, “Tracking without bells and whistles,” 2019, arXiv:1903.05625. [Online]. Available: http://arxiv.org/abs/1903.05625

[58] M. Keuper, S. Tang, B. Andres, T. Brox, and B. Schiele, “Motion segmentation & multiple object tracking by correlation co-clustering,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 1, pp. 140–153, Jan. 2020.

[59] P. Chu, H. Fan, C. C. Tan, and H. Ling, “Online multi-object tracking with instance-aware tracker and dynamic model refreshment,” in Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV), Jan. 2019, pp. 161–170.

[60] L. Chen, H. Ai, C. Shang, Z. Zhuang, and B. Bai, “Online multi-object tracking with convolutional neural networks,” in Proc. IEEE Int. Conf. Image Process. (ICIP), Sep. 2017, pp. 645–649.

[61] S.-H. Bae and K.-J. Yoon, “Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 1218–1225.

[62] L. Ma, S. Tang, M. J. Black, and L. V. Gool, “Customized multi-person tracker,” in Proc. Asian Conf. Comput. Vis. Cham, Switzerland: Springer, Dec. 2018.

[63] H. Sheng, J. Chen, Y. Zhang, W. Ke, Z. Xiong, and J. Yu, “Iterative multiple hypothesis tracking with tracklet-level association,” IEEE Trans. Circuits Syst. Video Technol., vol. 29, no. 12, pp. 3600–3672, Dec. 2019.

[64] C. Ma et al., “Trajectory factory: Tracklet cleaving and re-connection by deep siamese bi-GRU for multiple object tracking,” in Proc. IEEE Int. Conf. Multimedia Expo (ICME), Jul. 2018, pp. 1–6.

[65] H. Sheng, Y. Zhang, J. Chen, Z. Xiong, and J. Zhang, “Heterogeneous association graph fusion for target association in multiple object tracking,” IEEE Trans. Circuits Syst. Video Technol., vol. 29, no. 11, pp. 3269–3280, Nov. 2019.

[66] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro, Online Multi-target Tracking with Strong Weak Detections. Cham, Switzerland: Springer, 2016.

[67] R. Henschel, Y. Zou, and B. Rosenhahn, “Multiple people tracking using body and joint detections,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops, 2019, p. 0.

[68] R. Henschel, L. Leal-Taixe, D. Cremers, and B. Rosenhahn, “Fusion of head and full-body detectors for multi-object tracking,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jun. 2018, pp. 1428–1437.

[69] M. Keuper, S. Tang, B. Andres, T. Brox, and B. Schiele, “Motion segmentation & multiple object tracking by correlation co-clustering,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 1, pp. 140–153, Jan. 2020.

Jun Xiang received the B.S. and M.S. degrees from the College of Electronics and Information Engineering, South-Central University for Nationalities, in 2008 and 2012, respectively, and the Ph.D. degree in pattern recognition and intelligent systems from the Huazhong University of Science and Technology, Wuhan, China. She is currently an Assistant Professor with the Department of Electronics and Information Engineering, South-Central University for Nationalities, Wuhan. Her research interests are deep learning, graphical models, image and object segmentation, and object tracking.

Guohan Xu received the B.S. degree from the Department of Electronics and Information Engineering, South-Central University for Nationalities, Wuhan, China, in 2017, where he is currently pursuing the M.S. degree with the Department of Electronics and Information Engineering.

Chao Ma received the Ph.D. degree from Shanghai Jiao Tong University, in 2016. He was a Senior Research Associate at the School of Computer Science, The University of Adelaide, from 2016 to 2018. He is currently a tenure-track Assistant Professor with AI Institute, Shanghai Jiao Tong University, Shanghai, China. His research interests include computer vision and machine learning. He was sponsored by the China Scholarship Council as a visiting Ph.D. student in University of California at Merced from Fall 2013 to Fall 2015.

Jianhua Hou received the B.S. degree in optical engineering from the Beijing Institute of Technology in 1985, the M.S. degree in optical engineering from the University of Electronic Science and Technology of China in 1987, and the Ph.D. degree in pattern recognition and intelligent systems from the Huazhong University of Science and Technology, Wuhan, in 2007. His research interests are pattern recognition and computer vision.