Survey on ”Online Multi-Target Tracking Using Recurrent Neural Networks”

Yifan Chen
yifan.chen@stud.uni-goettingen.de

***Abstract*—This survey is based on the paper [3], which presents a recurrent neural network(RNN) based multiple object tracking(MOT) system. Compared with another existing MOT systems, this system mainly focuses on achieving the balance between efficiency and timeliness. A revolutionary innovation of this paper is that this is first approach to apply RNN in MOT task. This system also has following advantages: a) This is a model-free method, which mean it is not necessary to know prior knowledge. b) The system applies online algorithm to enable short time delay. c) The system combines RNN and lorg short-term memory to build suitable dynamic model. This survey will firstly give a brief introduction to background information. And then analyses four components of this MOT system. Finally, this survey will compare its performance with with another previous method to draw a short conclusion and point out the possible future work.**

***Index Terms*—Multiple object track, Recurrent neural network, Long Short-Term Memory, online learning**

I. Introduction

A. Multiple object tracking

Lots of mature algorithms applied in single object tracking. For example, Kalman filter, extended Kalman filter, particle filter and Mean shift algorithm. These technologies ensure high accuracy and precision in single object tracking. However, tracking a single object will not satisfy the requirement for today. Hence, multiple object tracking system was presented. MOT is now a hot topic in computational vision field. It aims to track simultaneously multiple object and get their trajectory. However, multiple object tracking faces following challenges [4]:

- *Identify unique ID to every object.* Since MOT needs to tracking multiple objects, it is difficult to distinguish between multiple objects. This issue usually happens when objects have high Similarity. A typical scenario is football match. Every football player wears same color polo shirt and his face feature will become obscure when they are doing high-velocity motion. ID switch usually happen in this case.
- *Identify multiple object types.* Most MOT tasks are tracking pedestrians in different scenarios. However, MOT cannot always detect only one type object. A common scenario is street monitoring. MOT system will needed to classify a object from label set [pedestrian, car, bicycle and so on] and maybe need to do a intelligent classification [5]. But a car with a driver may have the whole feature from ”car” label and partial feature from ”pedestrian” label. In this case, the definition of object somehow is ambiguous so MOT need to learn a suitable feature detection algorithm.
- *Track initiation and termination.* For a successful MOT system, it is importance to assign ID to new object when it access receptive field. And when object disappears from receptive field, MOT system needs to identify object leaves temporarily or forevermore. System should be robust against temporary leave and recover tracking when object is back to receptive field. And if object leave receptive field forever, systems should terminate tracking to save resource for new object.
- *Interaction and occlusion between multiple objects.* Since there are various objects are tracked in receptive field and everyone is represented with a bounding box. Issues happen when multiple bounding boxes get too close or even overlap with each other. In this case, system maybe consider wrong object so it will badly reduce system accuracy.
- *Potential dimension disaster.* Since system needs to store information about object. The size of store matrix has a strong impact on final performance. The number of tracking object number and duration of tracking time are two dominant parameters to decide the memory and CPU usage. In order to save computational resource, it is necessary to apply some optimization algorithm. For example, a dynamic tracking number and adaptive tracking time duration strategy may help in this case.

- *Multi-camera Multi-Object Tracking.* This is new challenge since more and more sensors are added to tracking system. The bottleneck is how to build the association between local information from single camera and global information from whole system [6]. Meanwhile, in consideration of low delay, this task should be completed in a given limited time.

These challenges are main bottleneck in multiple objects tracking field and now, more and more new and advanced algorithms are designed and applied to solve that. This survey will focus on new approach, which combines recurrent neural network firstly with online multiple objects tracking task [3]. This paper is based on the paper presented by Ondruska [11] [12] and it has following advantages: 1) This paper gives a model-free approach, which means prior knowledge it is not longer necessary. RNN will automatically train a dynamic model and the model size could be modified to fit in data size. Theoretically for arbitrary data size, RNN model could always fit it with a suitable network size. 2) RNN is a suitable deep learning model for analyse temporal type data. The data set in this paper is some videos data, and these data could be considered as the collection of frame time sequence. One important characteristic is that there exists strong temporal dependency between these frame. Traditional neural network, for example convolutional neural network, focus more on spatial dependency but not temporal dependency. 3) This paper apply online tracking but not offline tracking to avoid time redundancy. Online method means system are using current system state to product next system state. Offline method, or called "batch method", uses a small batch of system state to product next system state [13]. Obviously offline method utilizes more system so get higher accuracy in expectation. However, offline method has a time redundancy so online method is more suitable for a timely system. MOT system will usually contains three steps: predict step, data association step and update step. This paper uses RNN for predict step anmd LSTM for update step. A short introduction will be presented in following two subsections.

II. Multi-Target Tracking with RNNs

A. Overview

This is the topological graph of this paper1: There are four main stages: prediction stage, measurement stage, data association stage and update stage. And here we firstly make clear the notations. $x^t \in R^{N*D}$ represents the position matrix of all targets at one time(e.g one frame). N is the number of interacting targets simultaneously in one frame. And D is the order of position vector. This paper uses bounding box coordinates (x, y,w, h), so D = 4. For x^t, the 4 numbers of entire row represent the coordinates of middle point (x,y) and the width(w) and hight(h) of bounding box. And the ith column represents tth target. $z^t \in R^{M*D}$ represents the measurement matrix of possible detected targets at one time. M is the maximum number of detections targets at one time. The assignment probability matrix $A \in [0,1]^(N*(M+1))$ represents for each target to distribution of assigning individual measurement to that target [3]. $\varepsilon \in [0,1]$ represents the existence probability that indicate whether a target initialize or terminate.

B.

C. Measurement Stage

Although this part is hidden in this paper, here will give some details about the implementation –deformable part model algorithm. This algorithm is based on spring deformation model and developed from histogram of oriented gradient algorithm [1]. The basic idea of DPM is to find the global feature of target and match it with the detected target. And the processing are follow below steps [2]:

1) Do the preprocessing. Turn the colourful image to grey image. And do the global normalization to decrease the influence of illumination.
2) Divide the whole image into cells (8*8 pixel), and then calculate the magnitude and orientation of gradients of every cells. Build two histograms of each cells about 9 orientation bins (0' 180') and 18 orientation bins (0' 360'). And then add the row and column to gain a feature vector with 31 dimensions.
3) Create three suitable convolutional kernal functions for do the convolution operation with image2. And calculate the final score of current receptive field.
4) Do the softmax function to gain the receptive field which gets the highest match score and mark the target with bounding box. Output the bounding box coordinate as position vector.

The final output resulti is z^t.

D. Recurrent neural network

Since the first propose of computational neural model, neural network have developed for a long time. Various neural

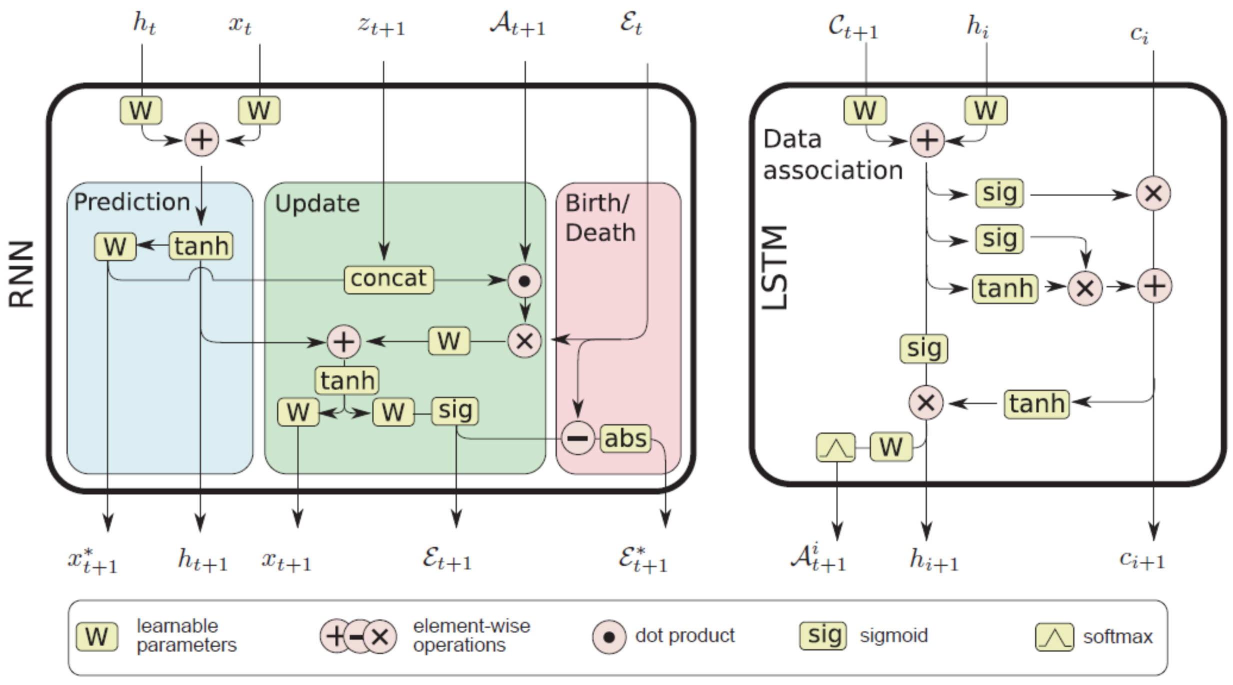


Figure 1: A overview of the structure. RNN responsible for prediction and update. And LSTM network responsible for the combinational problem, data association.

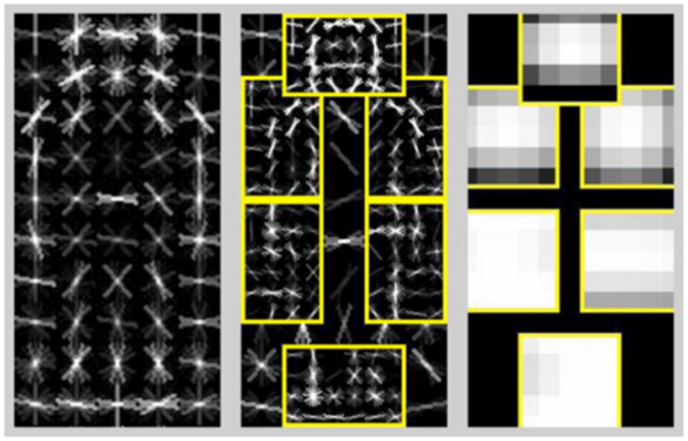


Figure 2: Three kernal for DPM. Root filter(left) to get the match score of global shape. Part Filter(middle) to get the match score of component model. Spring filter(right) to get the shift score by the distance of component.

network types are presented for vast machine learning tasks. Undoubtedly. recurrent neural network is one of most typical network network. RNN is use circulation form to handle data: Dates are firstly separated into fixed length batch and feed to RNN. Dates are usually need to obey a serious rule: early date first, to ensure the temporal dependency of date series. The most successful learning tasks at present are hand-writing recognition and automatic translation. RNN uses BPTT(Back Propagation Through Time) to modify its hidden system state and the temporal dependency is stored in hidden system state [14].The hidden system state H_t and system output Y_t at time t should be calculated as following [15]:

$$H_t = \sigma(I_t * W_{weights} + W * H_{t-1}) \tag{1}$$

$$Y_t = softmax(H_{weight} * H_t) \tag{2}$$

The activation function σ can be Tanh, Relu, Sigmoid and so on. Some advanced RNN model, so called ”Encode-Decode”, will create two RNN networks as encoder and decoder. Attention layer will also added to improve its accuracy. RNN are suitable to handle predict task, since we consider the precondition for prediction is Markov Assumption: the current system state is only depends on the previous system state. And the transfer processing could be predicted by transfer Matrix (hidden system state H_t in RNN). Although RNN can store temporal dependency, but the hidden system state can only store "close history information". The reason is that when RNN use BPTT to update the hidden system state, the influence from very beginning will decay and finally does not contribute to the output. This phenomenon is called "Gradient Vanishing" [7] and it prevents RNN to use global history information. Another

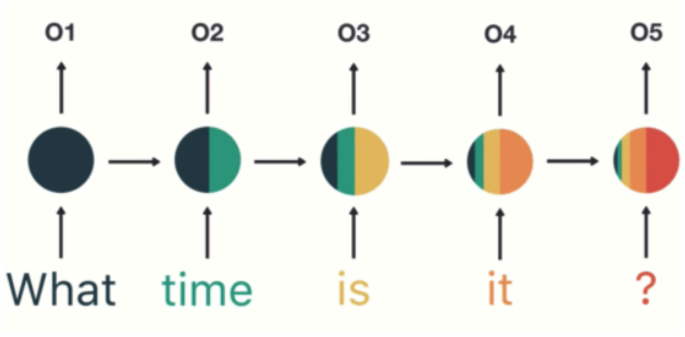


Figure 3: The colour of each node represent the influence of previous nodes. Obviously the output at t= 5 get little influence from input at t= 1 [7]

rare case is "Gradient Exploding" but will not cover in this survey. But for MOT system, access global history information will be important. For example, in data association task it will be helpful if system can access the target position information a hour ago. So as the result, this paper use RNN's advance form LSTM for data association task.

E. Long-Short Term Memory

Long-short term memory, also called LSTM, is a developed form of RNN. LSTM have same structure as RNN but beside, it adds two more components: forget gate and remember gate. Compared with RNN, LSTM has two system states, hidden stateh_t and cell state c_t [9]. These states can be calculated as following:

There are main three processing stages inside LSTM:

1) Forget stage: In this stage, system will operate "selective forgotten" to forget important information. Specifically, system will calculate z^f ("f" represents "forget") as forget gate controller to control forgotten of last system state c^{t-1}.

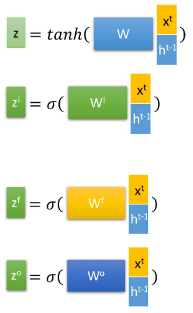

z = tanh(W x^t h^{t-1})
z^i = σ(W^i x^t h^{t-1})
z^f = σ(W^f x^t h^{t-1})
z^o = σ(W^o x^t h^{t-1})

Figure 4: z can be get by concat current system input x^t and last hidden system state h^{t-1}. z^o, z^f and z^i represent gate-control states. [9]

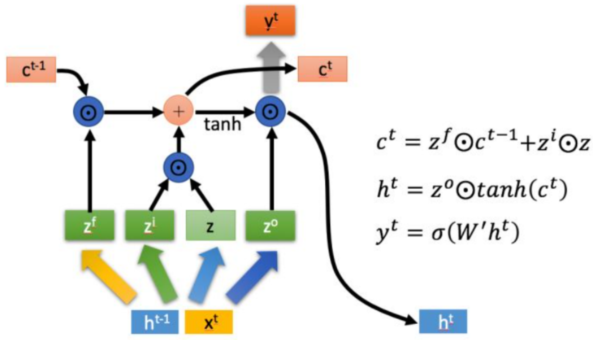


Figure 5: Topological graph of LSTM. ⊙ is hadamard product. [9]

2) Remember stage: In this stage, similar to forget stage, system will operate "selective memory" to remember important information. Specifically, system will calculate z^i ("i" represents "information") as remember gate controller to control how much current system input $x^(t)$ should be remembered.
3) Output stage: In this stage, z^o ("o" represents "output") will decide what will become the current system output.

Unlike RNN just have monotonous timely memory method, LSTM have multiple flexibly memory method. Due to this character, LSTM can handle memory which involve long memory.

F. Predict Stage

Firstly a dynamic RNN model need to trained to predict the next position state. In training phase, a labeled date set will be feed to model. The input vector is x^{t-1} ,which represents the position state of last moment. And the position state of current moment x^t is depends solely on the x^{t-1} and system hidden state h^t as shown in figure 1. The loss function is given as mean square error between true value and measurement value:

$$Loss_{Prediction} = \frac{\lambda}{ND} \sum \|x^* - \widetilde{x}\|^2 \tag{3}$$

After training, every time given a position state x^t, system will predict the next position x^{t+1}.

G. Data Association Stage

When given a predict position, system will calculate the Euclidean distance between the predicted position and measurement position. Show in the form of pairwise-distance matrix $C \in R^{N*M}$, where $C_{ij} = \|x^i - y^j\|^2$ represents the distance between i_{th} prediction position and i_{th} measurement position. This pairwise-distance matrix will feed to LSTM network as well as two hidden system states h^i and c^i. And then output the assignment probability matrix $A \in [0,1]^{N*(M+1)}$. The training loss is:

$$Loss_{DataAssociation} = -log(A_{i\tilde{a}}) \tag{4}$$

where $\tilde{a}$ is the correct assignment and A_{ij} is the target i to measurement j assignment probability.

H. Update Stage

Here in update stage, system uses the conjoint vector from measurement z^t value and predicted value x^t, to multiple the assignment probability matrix A_t and existence probability vector ε_t to calculate ε_{t+1} and update position state x^{t+1}. Also, a parameter ξ is trained to represent the existence possibility of target. And ξ_t represent the existence possibility of all target at the given time t. This factor is used for predict target initiation and termination. In this paper, the threshold is set as $\xi_{existence} = 0.6$. And this is the demo figure in the training phase6:

In training phase, the loss function is given as mean square error between true value and predict value and the loss function of ξ:

$$Loss_{Update} = \frac{k}{ND} \sum \|x - \widetilde{x}\|^2 + v * Loss_{\xi} + \phi \tag{5}$$

Where ϕ is the smoothness factor to help system overcome the "Hard decision" issue [9] and the loss function of ξ is given as binary cross entropy.$\tilde{\xi}$ is the true existence possibility of tracking target:

$$Loss_{\xi} = \tilde{\xi} * \log \xi + (1 - \tilde{\xi}) * \log 1 - \xi \tag{6}$$

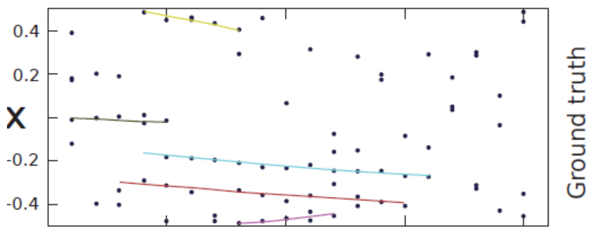


Figure 6: Simulation result on a 20-frame long synthetic sequence with clutter (x-coordinate vs. time). [9]

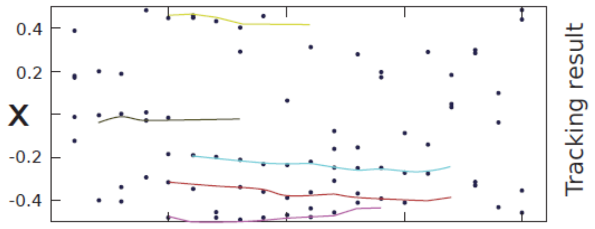


Figure 7: Reconstructed trajectories during experiment. [9]

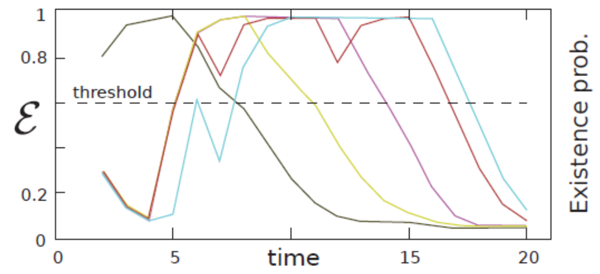
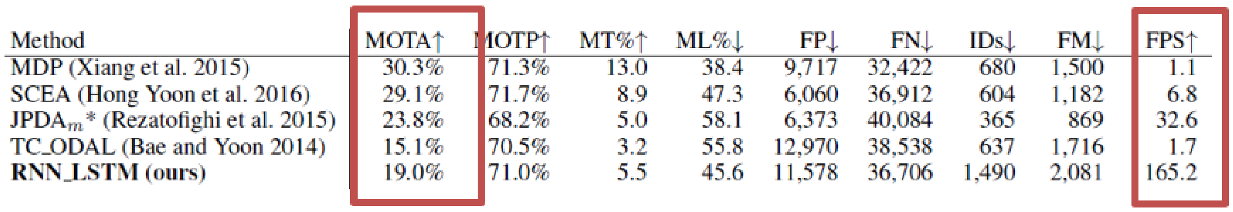


Figure 8: The ξ represents the existence probability of each target. Each colour represent one target. When the system consider the ξ of one target is bigger than 0.6, the system start tracking its trajectory. And when the ξ is less than 0.6, system will terminate the tracking. [9]

III. Conculsion

A. Performance

Firstly this paper compares its performance with three baselines method. Kalman-HA method, which combines Kalman filter with bipartite matching solved via the Hungarian algorithm. Kalman-HA2, additionally adds heuristics to remove false tracks [9]. $JPDA_m$ is the full joint probabilistic data association approach [10]. This paper admires these methods do have advantages in some aspects, e.g. make use of vision feature, likes optic flow to improve the data association performance. However, the authors also argues that these feature is not general in cell tracking and animal tracking. And two of them are offline method which sacrifice the system timeliness. After that, this paper also compares itself with state-of-art methods. According to comparing with some state-of-the-art method, this paper have proves that the combination of RNN and LSTM is a power tool to solve multiple objects tracking



Method	MOTA↑	MOTP↑	MT%↑	ML%↓	FP↓	FN↓	IDs↓	FM↓	FPS↑
MDP (Xiang et al. 2015)	30.3%	71.3%	13.0	38.4	9,717	32,422	680	1,500	1.1
SCEA (Hong Yoon et al. 2016)	29.1%	71.7%	8.9	47.3	6,060	36,912	604	1,182	6.8
JPDA$_m$* (Rezatofighi et al. 2015)	23.8%	68.2%	5.0	58.1	6,373	40,084	365	869	32.6
TC_ODAL (Bae and Yoon 2014)	15.1%	70.5%	3.2	55.8	12,970	38,538	637	1,716	1.7
RNN_LSTM (ours)	19.0%	71.0%	5.5	45.6	11,578	36,706	1,490	2,081	165.2

Figure 9: The experiment result compared with state-of-art methods in same data set. [9]

problem. This is the first RNN combine with MOT but we believe there will be more and more new neural networks type and new strategies will applied in this field. After that, this paper also give a strong evidence that online tracking method can achieve fair balance. So according to the discussion above, we can draw a conclusion that two breakthrough are 1) Achieve the balance between accuracy and efficiency. 2) Two orders of magnitude faster than state-of-art method. The result is shown in 9

B. Future Work

Of course this paper has its limitation, so here we point out some field which could be give more research.

- Replace RNN neural unit. The ”gradient vanishing” is just one disadvantage of RNN, nowadays some strategy, likes add attention layer, have applied in this field. However it will be better if we can use LSTM or GRU, or even full attention component to replace RNN totally, we believe such methods will increase accuracy.
- Decrease training period. Since RNN are feeded with temporal date so it have to handle the data lineally. It is impossible to speed up training processing by just add more computational resources likes CPU and GPU so far. We hope someday it will have breakthrough on this field, so we can apply parallel method in RNN.
- Achieve better accuracy. Since this paper have gain a great success in MOTA norm, we think it is worth if the system can use a small batch frames for prediction and data association. The perfect state is this method is faster and more accurate than the state-of-art method.

References

References

[1] Fischler, M. A., & Elschlager, R. A. (1973). The representation and matching of pictorial structures. IEEE Transactions on computers, (1), 67-92.

[2] Felzenszwalb, P. F., McAllester, D. A., & Ramanan, D. (2008, June). A discriminatively trained, multiscale, deformable part model. In Cvpr (Vol. 2, No. 6, p. 7).

[3] Milan, A., Rezatofighi, S. H., Dick, A., Reid, I., & Schindler, K. (2017, February). Online multi-target tracking using recurrent neural networks. In Thirty-First AAAI Conference on Artificial Intelligence.

[4] Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W., Zhao, X., & Kim, T. K. (2014). Multiple object tracking: A literature review. arXiv preprint arXiv:1409.7618.

[5] Milan, A., Leal-Taix¨ı, L., Reid, I., Roth, S., & Schindler, K. (2016). MOT16: A benchmark for multi-object tracking. arXiv preprint arXiv:1603.00831.

[6] Dockstader, S. L., & Tekalp, A. M. (2001). Multiple camera fusion for multi-object tracking. In Proceedings 2001 IEEE Workshop on Multi-Object Tracking (pp. 95-102). IEEE.

[7] Madhu Sanjeevi. (2018,Jan). Chapter 10: DeepNLP - Recurrent Neural Networks with Math.

[8] Hung-yi Lee. (2017,March). Conditional Generation by RNN & Attention

[9] Greff, K., Srivastava, R. K., Koutn¨ªk, J., Steunebrink, B. R., & Schmidhuber, J. (2016). LSTM: A search space odyssey. IEEE transactions on neural networks and learning systems, 28(10), 2222-2232.

[10] Hamid Rezatofighi, S., Milan, A., Zhang, Z., Shi, Q., Dick, A., & Reid, I. (2015). Joint probabilistic data association revisited. In Proceedings of the IEEE international conference on computer vision (pp. 3047-3055).

[11] Kumar, A., Irsoy, O., Ondruska, P., Iyyer, M., Bradbury, J., Gulrajani, I., ... & Socher, R. (2016, June). Ask me anything: Dynamic memory networks for natural language processing. In International conference on machine learning (pp. 1378-1387).

[12] Ondruska, P., & Posner, I. (2016, March). Deep tracking: Seeing beyond seeing using recurrent neural networks. In Thirtieth AAAI Conference on Artificial Intelligence.

[13] Duchi, J., & Singer, Y. (2009). Efficient online and batch learning using forward backward splitting. Journal of Machine Learning Research, 10(Dec), 2899-2934.

[14] Gruslys, A., Munos, R., Danihelka, I., Lanctot, M., & Graves, A. (2016). Memory-efficient backpropagation through time. In Advances in Neural Information Processing Systems (pp. 4125-4133).

[15] Sherstinsky, A. (2018). Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. arXiv preprint arXiv:1808.03314.