AOL: Adaptive Online Learning for Human Trajectory Prediction in Dynamic Video Scenes

Manh Huynh
manh.huynh@ucdenver.edu
Gita Alaghband
gita.alaghband@ucdenver.edu

Department of Computer Science and Engineering
University of Colorado Denver, USA

Abstract

We present a novel adaptive online learning (AOL) framework to predict human movement trajectories in dynamic video scenes. Our framework learns and adapts to changes in the scene environment and generates best network weights for different scenarios. The framework can be applied to prediction models and improve their performance as it dynamically adjusts when it encounters changes in the scene and can apply the best training weights for predicting the next locations. We demonstrate this by integrating our framework with two well-known prediction models: LSTM [1] and Future Person Location (FPL) [22]. Furthermore, we analyze the number of network weights for optimal performance and show that we can achieve real-time with a fixed number of networks using the least recently used (LRU) strategy for maintaining the most recently trained network weights. With extensive experiments, we show that our framework increases prediction accuracies of LSTM and FPL by 17% and 28% on average, and up to 50% for FPL on the worst case while achieving real-time (20fps).

1 Introduction

The problem of predicting future human movements in dynamic video scenes presents interesting challenges in developing a trajectory prediction network. Highly complex neural-based networks [1, 19, 22] are trained for certain scenarios first and then applied in the test phase for the purpose of predicting future locations using the best fixed network weights obtained from the extensive training. These networks learn from human past movements using feature cues such as past human locations, camera motions, human poses, and human social interactions using large datasets during the training and are expected to generalize to new testing videos/scenes. This approach often performs well for scenarios that have been similarly encountered during the training phase. However, they do not predict with high accuracy new circumstances encountered in dynamic video sequences due to sudden context changes as in camera movement, angle, speed, crowd behavior, and scenes (e.g. streets, markets, parking lots, etc.).

To demonstrate this, one needs to look at predictions at a finer level of granularity than the average results that are normally reported. To motivate the discussion, we trained two

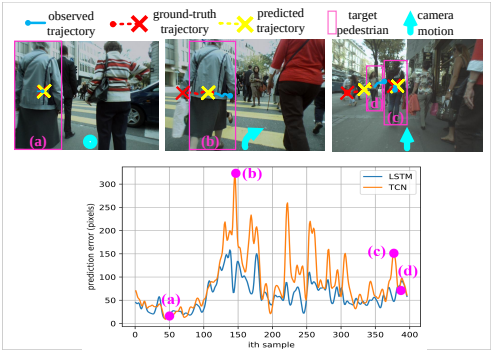

Figure 1: The trained networks (e.g. FPL [22], LSTM [1]) are unable to generalize well enough when there are rapid context changes such as stable camera motions (a), rapid sudden camera motions (b), or scenes with a mix of human movement directions (c) and (d).

well-known prediction networks representing different spectrum of existing methods, Long Short-term Memory (LSTM) [1] and Future Person Location (FPL) [22] on a large Locomotion dataset [22] and tested them on a new video from ETH dataset [2]. The weights obtained at the conclusion of the training phase of each network remained fixed during testing. LSTM learns motion patterns of a person from their past positions in a sequential manner. The motion pattern encoded in LSTM hidden states can be used to predict a pedestrian's future positions. On the other hand, FPL [22] uses Temporal Convolutional Neural Network (TCN) [3] as encoders and a decoder. The encoders encode features such as human locations, scales, human poses, and camera motions extracted from observed frames. These encoded features are then concatenated (channel-wise) and decoded into predicted future locations. An advantage of using TCNs over the LSTM network is that it can encode multiple observed features in parallel, allowing more features to be incorporated with faster performance. Conversely, LSTM requires less memory and performs slightly better than TCNs for prediction in long sequences [3].

Figure 1 shows the future location prediction results in final displacement error in pixels (FDE), where each sample corresponds to a pedestrian's future trajectory in the next one second or 10 frames. From the variations plotted, we observe that neither network can perform consistently well across all samples. While both networks predict with high accuracy (lower prediction error) when the camera motions are stable (scenario a); they perform poorly when there are abrupt camera motions (scenario b). Another context change that the two networks do not capture well is shown in scenarios c and d when the camera is steady (camera consistently moves forward), predicting the future locations of pedestrians moving-forward (scenario c) is harder than pedestrians moving-away (scenario d).

To address this problem, we believe that a network should have the ability to adapt its weights to different contexts. The first approach we explore in this paper, base adaptive online learning framework (B-AOL), is to continue having the ability to train after the initial training phase. B-AOL consists of a master network and a slave network. Both share the same prediction network architecture such as LSTM [1] or FPL [22]. While the master network continuously trains and its weights change as it encounters new contexts, the slave network uses a copy of master's pre-trained weights (last training stage) to test the incoming samples. The weights for the slave network are switched to that of the master network upon encountering new contexts where the slave predictions are inaccurate. This process ensures that the master network is always updated with contexts of recent samples to achieve higher prediction accuracy for the next samples as there is a high probability that nearby samples possess similar temporal dynamics. However, this approach can fail when several abrupt context changes are encountered close to each other. In such a case, B-AOL will not be able to capture all scenarios due to inadequate training data. We address this issue by designing an adaptive online learning model (AOL) to efficiently adapt to dynamic scene context changes

by generating and maintaining one master network that continues to train, and multiple slave networks, each with the best trained set of weights matching a specific context encountered during the testing process up to the current point. This allows us to cope with the context changes in the next testing samples as there is a high probability that the new sample contexts will be similar to one of our previously generated best-performing network weights. The challenge arises because the number of prediction network weights increases linearly with the number of new contexts encountered in the streaming video resulting in large memory usage and slow execution time. To handle this problem, we maintain a Least Recently Used (LRU) network replacement policy where we can control the number of slave networks. The LRU enables us to keep the most updated slave networks in the list by continuously updating their weights with recent master network's weights replacing those that have not performed well (used) for the longest time. More importantly, by controlling the number of networks, our framework is able to perform in real-time.

Our adaptive online learning framework is flexible in that it can incorporate different types of prediction networks and improve their prediction accuracy (Section 4). We conduct extensive experiments integrating our framework with two state-of-the-art prediction network models on two dynamic video sequence datasets: First-person Locomotion [22] and ETH [6] and show significant accuracy improvements over the stand-alone prediction networks on these datasets (Section 5).

2 Related Work

Human Trajectory Prediction in Dynamic Video Scenes. Most of the recent work [1, 21, 22] rely on recurrent neural networks (RNNs) [14], temporal convolutional neural networks (TCNs) [3], or their variants [10, 23] to predict human future locations. Furthermore, several attempts have been made to model human-scene interactions [4, 11]and/or human-human interactions [10, 22]. However, the aforementioned methods only perform well in static scenes, where bird's eye views are given. Forecasting human future location in dynamic scenes is more challenging due to the abrupt camera motions, vast varieties of human poses and dynamic scene change from one place to another. Some [19, 22] have attempted to model the camera motions and human poses in prediction networks. They focus on improving network architectures and crafting input features. However, with the limited training data, these networks do not generalize in different contexts during testing. To tackle this problem, we propose a novel adaptive online learning framework to boost the accuracy of the existing prediction models.

Online Network Adaptation. To the best of our knowledge, there is no existing online adaptation work for trajectory prediction in dynamic scenes. The recent research in unsupervised domain adaptation [8] and meta-learning [7] possibly presents the closest ideas to our work, but they are fundamentally different. Meta-learning [7] presents the a concept of learning-to-learn, which learns specific model parameters for a given domain. However, a small ground-truth data for each domain must be given, which limits its applicability to real-time and practical applications. Common approaches [8, 9, 20] for unsupervised domain adaptation minimize domain discrepancy between source and target domain from raw input data without the need for ground truth data in target domain. Though, metadata (e.g. weather conditions, day/night,...) must be available. Due to this assumptions, it is non-trivial to apply the methods in meta-learning and unsupervised domain adaptation for human trajectory prediction in dynamic scenes. There are some online adaptation methods, which

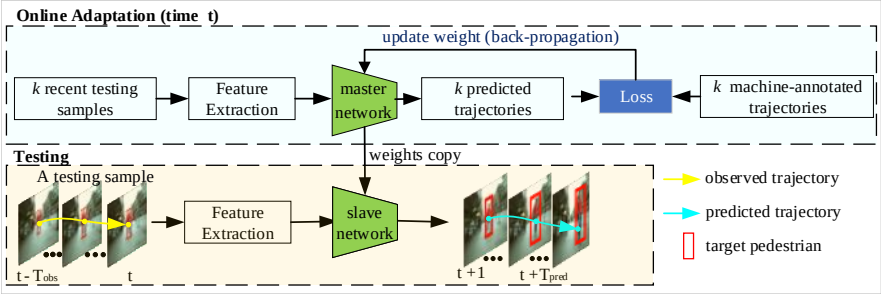

Figure 2: Base online adaptation framework (B-AOL) for human trajectory prediction.

are customized for applications such as object tracking [16], video segmentation [21], robot motion planning [15]. But, it is not straightforward to adapt these networks for predicting human future locations due to network architecture differences.

3 B-AOL: Base Online Adaptation Framework.

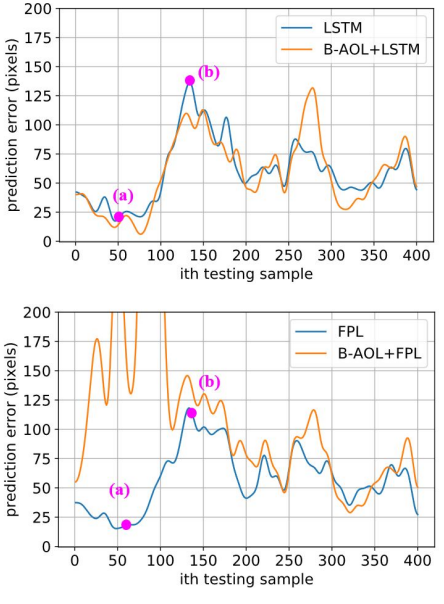

Figure 3: The prediction error (lower is better) of base online adaption framework (B-AOL) (with k =1) compared with no-adaptation networks on each testing sample using LSTM network (top graph) using FPL network (bottom graph).

The base online adaption framework (B-AOL) adapts a prediction network (LSTM and FLP in this paper) to the context changes after an initial training phase by updating the network's weights using k recent testing samples. Figure 2 depicts the overview of B-AOL for human trajectory prediction in dynamic streaming video. The framework consists of a master network and a slave network.

In the online adaptation phase, the master network's weights are first initialized with the best-trained prediction network' s weights on a large dataset. At each time t of a new video being tested, the master network is trained using k recent testing samples. To train the master network, we calculate a mean square error (MSE) loss between the recently predicted trajectories and the high confident machine-annotated trajectories generated from a human tracking algorithm (e.g. Track-RCNN [21]. The loss's gradient is backpropagated to update the master network's weights using mini-batch gradient descent [17]. Its trained weights are then copied to the slave network for testing the upcoming samples. Note that using the high confident machine-annotated

data does not cause any error drifts during training and has been efficiently used as augmented data [21].

In the testing phase (prediction phase), the slave network predicts future locations of the target pedestrian (in the red bounding box) in the next T_{pred} frames by observing the past human trajectories and other features extracted from T_{obs} frames. For evaluation, the predicted future locations are compared (using MSE) with ground-truth trajectories. The details of feature extraction and evaluation metrics are discussed in Section 5. The base online adaptation framework poses two sets of challenges:

(1) B-AOL cannot effectively adapt to sudden and unexpected changes in different scene contexts. Figure 3 shows examples of this. We integrated the LSTM and FLP prediction networks with our B-AOL framework and repeated the experiment presented in Section 1 (B-AOL+LSTM and B-AOL+FPL). The graphs show FPL performs better than LSTM in general; but also that B-AOL+LSTM and B-AOL+FPL do not show improvements in either case when there are abrupt camera motions (Figure 3 , scenario b). Additionally, we observe that B-AOL+FPL's prediction accuracy worsens compared to FPL even when the camera motion is stable at the beginning of testing video due to the domain shift problem [18]. This usually happens when the data (features distributions from the training datasets are different from the test datasets.

(2) To improve the prediction accuracy of B-AOL, one can increase the number of k recent testing samples and use a higher number of training epochs. However, due to the runtime constraints, this approach is unrealistic for real-time applications. The study of how sample sizes and training epochs impact the prediction accuracy and processing time are presented in the ablation studies in Section 5.

4 AOL: Adaptive Online Learning Model.

To address the shortcomings of the B-AOL framework, we propose a novel adaptive online learning framework (AOL). The key idea is to maintain a master and a list of multiple slave prediction networks $S = \{s_0, s_1 \ldots, s_n\}$, where s_i performs best in a specific context i. Having $n+1$ slave networks enables AOL to predict with higher accuracy for $n+1$ different scene contexts. Thus, when the new scene contexts are encountered, there is a probability that the new scene context will be similar to one of the previously learned $n+1$ contexts.

The AOL framework is depicted in Figure 4. We use the same online adaptation procedure as B-AOL for training the master network. The testing process works as follows:

- At time $t = 0$, we initialize the slave network list S with s_0, $S \leftarrow \{s_0\}$. At every time step, the trained master network's weights obtained from the recent testing samples are copied to the slave network s_0 (Figure 4, step 1). This ensures we always have a slave network (i.e. s_0) that is updated with the most recent context changes.

- To predict a target's future locations in the next T_{pred} frames, we extract features from T_{obs} frames. The extracted features are then input to all slave networks $s_0, s_1 \ldots, s_n$ (Figure 4, Step 2) to generate a list of predicted trajectories corresponding to the output of the slave networks. We note that the slave networks share the same network architectures, only their weights are different. The slave network that generates the best predicted trajectory is selected.

- To find the best predicted trajectory (Figure 4, step 4), we calculate the mean square error (MSE) between all predicted trajectories in the list with the machine-annotated

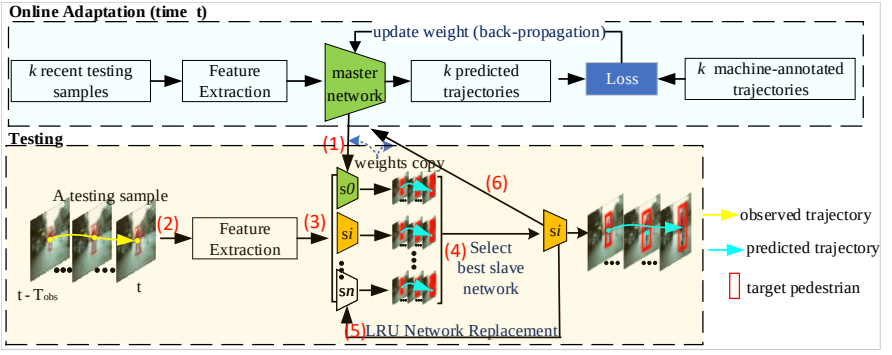

Figure 4: The overview of the proposed framework (AOL) for human future trajectory prediction.

trajectories generated from a human tracking algorithm (e.g. Track-RCNN [21]). The best predicted trajectory is the one with the lowest MSE. Using machine-annotated trajectories is appropriate in testing phase because one should not have access to ground-truth trajectories while testing.

- Lastly, the list of slave networks is updated using the following scenarios: (1) If the best-selected slave network $s_b = s_i$ with $i > 0$ then use prediction from this slave network. (2) If the best-selected slave network is s_0 (i.e. $s_b = s_0$). We save a copy of s_b into the slave network list using Least Recently Used (LRU) network replacement policy. Specifically, if the current number of slave networks p has not reached the pre-defined maximum capacity ($p < n$), we add this network as a new entry in the list: $s_{p+1} \leftarrow copy(s_0); S \leftarrow S \cup s_{p+1}$. Otherwise, we replace it with the least recently used (LRU) network in the slave network list. Using LRU not only we maintain slave networks with the most recent contexts, we also can control the total number of these networks and achieve real-time processing.

5 Experiments

5.1 Experiment Setups

Datasets. We use Locomotion [22] and ETH [6] datasets for evaluation. The Locomotion dataset is captured from pedestrians wearing a chest-mounted camera on busy streets in Japan. The dataset consists of 50,000 samples of 5,000 pedestrians in total. Each sample consists of $T = 20$ frames (10 frames for observation and 10 frames for prediction).
We also conduct experiments on ETH datasets [6], captured from robots' cameras. The ETH datasets consist of 4 video sequences with 9000 samples are extracted. All frames in both datasets are scaled to size $960x1280$ and frame per second (fps) is 20.
Evaluation Metrics. We evaluate our system using two commonly used metrics [1, 19, 22]: (a) average displacement error (ADE): mean square error over all locations of predicted and true trajectories; (b) final displacement error (FDE): mean square error at the final predicted and true locations of all human trajectories.
Feature extraction. We used the following process to extract the required features for FPL and LSTM. For each video sequence frame, we extract an 18 key-point pose (36-dimensional vector) per pedestrian using OpenPose [5]. The human location (2-dimensional vector) is set

at the middle hip of the pose, the body scale (1-dimensional vector) is calculated as the height of the human body from the neck to the middle hip. To calculate the camera motions (i.e. ego-motions), we use grid optical flows. First, the optical flow for each pixel is calculated using FlowNet2 [12]. We then divide a frame into 3x4 grid-cells and calculate the average optical flow of all pixels within a grid-cell. The resulting 24-dimensional optical flow vector is used to represent the camera motion at a given frame. During the pre-training process of a prediction network dataset, we use ground-truth human trajectories for calculating the prediction loss. While during testing, we use the machine-annotated trajectories.

Implementation details. We implemented our framework using PyTorch [13] deep learning framework. During the online adaptation phase, the number of training epochs is set to 3, the number of recent samples $k = 1$, the learning rate is 0.001. The maximum number of slave networks is 10. The analysis of these parameters is presented in Section 5.3. We use Adam [12] for network optimization. To generate the machine-annotated trajectory of each sample, we used Track-RCNN [21]. The network models are trained/tested on GPU Tesla P100-SXM2.

Datasets	Locomotion	ETH
	ADE/FDE	ADE/FDE
LSTM	50.31/ 89.26	30.01/51.34
B-AOL + LSTM	62.49/109.34	28.49/49.34
AOL + LSTM	**41.69/72.55**	**24.68/41.81**
FPL	43.25/77.25	47.78/78.46
B-AOL + FPL	57.23/99.23	91/112.36
AOL + FPL	**33.03/55.76**	**34.49/54.28**

Table 1: Quantitative results of AOL on two datasets: Locomotion and ETH.

Components	FDE
LRU vs random network replacement	54.28 vs 56.15
with/without adapting master network using the best slave network's weights	54.28 vs 70.75

Table 2: Contributions of framework components.

5.2 Quantitative results on Locomotion and ETH datasets

Results on the Locomotion dataset. We apply the 5-fold cross-validation protocol similar to [22]. The dataset is split into 5; the prediction networks (LSTM and FPL) are initially trained on four splits (pre-training phase); we then test/adapt them using AOL on the remaining split. The average ADE/FDE over 5 splits are reported in Table 1. Our framework AOL reduces the ADE/FDE of LSTM and FPL by 17.13%/12.34% and 28.25%/27.81%, respectively. We can see that FPL is a better prediction network for adaptation. With more input features used, stand-alone FPL also performs better than LSTM (43.25 vs 50.31 ADE). However, B-AOL worsens the prediction results of both LSTM and FPL models as expected.

Results on the ETH dataset. We tested AOL on a different dataset to demonstrates it can predict with higher accuracy in dynamic scenes and effectively adapt to domain changes. We train LSTM and FPL on the Locomotion dataset using 100 training epochs and continue adapting/testing it using AOL on the ETH dataset. The second column in Table 1 shows AOL improves ADE/FDE of LSTM and FPL by 17.76%/18.56% and 27.78%/30.81% respectively. Interestingly, we notice that FPL does not perform better than LSTM on ETH datasets. This is due to the missing human pose key-points of several pedestrians in the ETH videos. This scenario happens frequently for pedestrians up-close to the camera or too small when too far from the camera. However, AOL still effectively adapts FPL by improving its prediction accuracy in these scenarios.

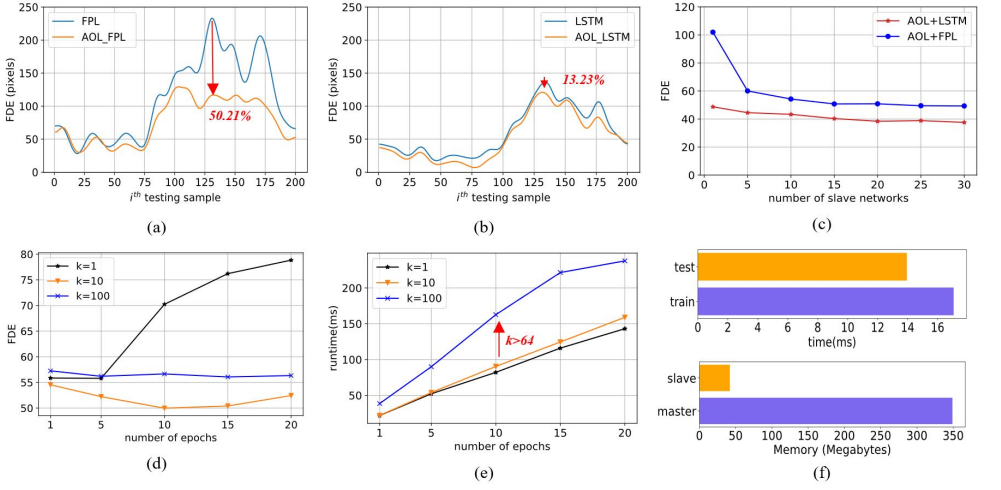

Figure 5: Analysis. (a,b) efficiencies of AOL under abrupt camera motion; (c) impact of number of slave networks; (d, e) impact of number of past samples and training epochs on FDE and runtime; (e) runtime and space complexities.

5.3 Analysis

In this section, we study in-depth the impacts of our design ideas and other parameters that impact on AOL's performances. The experiments are done using the ETH dataset.

Contributions of framework components. Table 2 shows the impacts of two design ideas of AOL: (a) using the least recently used (LRU) network replacement policy, and (b) copying the best slave network's weights to the master network for adapting the prediction/training to encounter new scene contexts. Compared to random network replacement policy, LRU maintains the most recent contexts. As a result, it generates more accurate predictions (lower FDE) compared to random network replacement (54.28 vs 56.15). Copying the best slave network's weights to the master network, so that the training can continue for the best and most recent prediction, also produces significantly higher prediction accuracy (54.28 vs 70.75 FDE). This is due to the similar temporal dynamics of samples in nearby frames.

Prediction under abrupt camera motions. Figures 5a and 5b show that AOL successfully handles abrupt camera motion scenarios. We observe that AOL reduces the prediction error (FDE) by up to 50.21% for FPL and 13.23% for LSTM for the worst-case scenario.

Impact of varying number of slave networks. Figure 5c shows the resulting FDE as we change the number of slave networks. As we increase the number of networks, FDE is consistently reduced until about 10 networks and seems to saturate when the number of networks reaches 20.

Time complexity. Figure 5f (top bar) shows the runtime of testing and training of the AOL+FPL given one testing sample and 10 slave networks. With only one sample used for training and 3 training epochs, the training time is 17 milliseconds (*ms*) per sample; while the test time of 10 slave networks is 13.91*ms* per sample. If the slave networks are tested in parallel, the testing time can be mitigated (to about 1.3*ms*). Overall, the total processing time per frame is 31*ms* sequentially, which is real-time processing in a video of

Figure 6: Qualitative results. AOL significantly improves the performance of FPL under various pedestrian movements and camera motions. Camera motions are stable on top figures, while there are abrupt changes in the bottom figures

framerate 20fps.

Space complexity. a master FPL network requires about 350 Megabytes (MBs), while each slave network requires a factor 7 less memory (Figure 5f, bottom bar). This is because the master network must store additional network parameters (i.e. gradients), while the slave network does not. However, the more slave networks used, the more memory is needed.

Impact of varying past samples (k) and training epochs(e). Figures 5d and 5e show FDE and runtime as we change the number of recent samples and training epochs. We fixed the number of networks to 10 and use the FPL prediction network. We note that when k is small, e (epoch size) should also be small. Otherwise, it is very easy to overfit to a specific context. On the other hand, when k is large, FDE does not improve because each slave network has an average performance in many contexts. Larger k and e also significantly affect the runtime (Figure 5e). Increasing k and e linearly increase the runtime. Interestingly, with batch-train size 64, we see a big jump in the runtime for $k = 100$. Thus, using large samples and training epochs are impractical for this real-time application. With 10 slave networks, we need to keep e smaller than 5 and k smaller than batch size to achieve real-time processing.

5.4 Qualitative results.

Figure 6 presents sample qualitative results showing improvements of AOL+FPL over standalone FPL on ETH datasets under various pedestrian movements: moving toward, away, across under different camera motions: stable and abrupt. When the camera is stable (top row), FPL and AOL+FPL produce comparable results. However, when there are abrupt camera motions, FPL falls short, while AOL+FPL maintains better prediction results.

6 Conclusions

In this paper, we presented a novel adaptive online learning framework (AOL) for human future location prediction in dynamic video scenes. AOL relies on the idea of a master network, which continuously trains to keep up with changes in scene contexts, and multiple slave networks, capable to produce the highly accurate predictions for the most recent n scene contexts encountered. AOL uses LRU to control the number of slave networks to achieve real-time and limit memory consumptions. AOL can integrate prediction network models and improve their performance in dynamic scenes. We presented this by integrating two well-known LSTM and FLP prediction networks with AOL and showed high adaptability and significant improvements in the prediction accuracy of these networks while achieving real-time performance.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016.

[2] Authors. Frobnication tutorial, 2006. Supplied as additional material `tr.pdf`.

[3] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[4] Federico Bartoli, Giuseppe Lisanti, Lamberto Ballan, and Alberto Del Bimbo. Context-aware trajectory prediction. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1941–1946. IEEE, 2018.

[5] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1812.08008*, 2018.

[6] Andreas Ess, Bastian Leibe, Konrad Schindler, and Luc Van Gool. A mobile vision system for robust multi-person tracking. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

[8] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014.

[9] Behnam Gholami, Pritish Sahu, Ognjen Rudovic, Konstantinos Bousmalis, and Vladimir Pavlovic. Unsupervised multi-target domain adaptation: An information theoretic approach. *IEEE Transactions on Image Processing*, 29:3993–4002, 2020.

[10] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2255–2264, 2018.

[11] Manh Huynh and Gita Alaghband. Trajectory prediction by coupling scene-lstm with human movement lstm. In *International Symposium on Visual Computing*, pages 244–259. Springer, 2019.

[12] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017.

[13] Nikhil Ketkar. Introduction to pytorch. In *Deep learning with python*, pages 195–208. Springer, 2017.

[14] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

[15] Anusha Nagabandi, Chelsea Finn, and Sergey Levine. Deep online learning via meta-learning: Continual adaptation for model-based rl. *arXiv preprint arXiv:1812.07671*, 2018.

[16] Eunbyung Park and Alexander C Berg. Meta-tracker: Fast and robust online adaptation for visual object trackers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 569–585, 2018.

[17] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[18] Swami Sankaranarayanan, Yogesh Balaji, Arpit Jain, Ser Nam Lim, and Rama Chellappa. Learning from synthetic data: Addressing domain shift for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3752–3761, 2018.

[19] Oliver Styles, Victor Sanchez, and Tanaya Guha. Multiple object forecasting: Predicting future object locations in diverse environments. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 690–699, 2020.

[20] Onur Tasar, SL Happy, Yuliya Tarabalka, and Pierre Alliez. Colormapgan: Unsupervised domain adaptation for semantic segmentation using color mapping generative adversarial networks. *IEEE Transactions on Geoscience and Remote Sensing*, 2020.

[21] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. *arXiv preprint arXiv:1706.09364*, 2017.

[22] Takuma Yagi, Karttikeya Mangalam, Ryo Yonetani, and Yoichi Sato. Future person localization in first-person videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7593–7602, 2018.

[23] Pu Zhang, Wanli Ouyang, Pengfei Zhang, Jianru Xue, and Nanning Zheng. Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12085–12094, 2019.