Towards View-Invariant Intersection Recognition from Videos using Deep Network Ensembles

Abhijeet Kumar*† Gunshi Gupta*‡ Avinash Sharma† K. Madhava Krishna‡

***Abstract*— This paper strives to answer the following question: Is it possible to recognize an intersection when seen from different road segments that constitute the intersection? An intersection or a junction typically is a meeting point of three or four road segments. Its recognition from a road segment that is transverse to or 180 degrees apart from its previous sighting is an extremely challenging and yet a very relevant problem to be addressed from the point of view of both autonomous driving as well as loop detection. This paper formulates this as a problem of video recognition and proposes a novel LSTM based Siamese style deep network for video recognition. For what is indeed a challenging problem and the limited annotated dataset available we show competitive results of recognizing intersections when approached from diverse viewpoints or road segments. Specifically, we tabulate effective recognition accuracy even as the approaches to the intersection being compared are disparate both in terms of viewpoints and weather/illumination conditions. We show competitive results on both synthetic yet highly realistic data mined from the gaming platform GTA as well as on real world data made available through Mapillary.**

I. INTRODUCTION

Recognizing an intersection from a different approach sequence or a sequence of viewpoints different from those seen before can be pivotal in various applications that include autonomous driving, driver assistance systems, loop detection for SLAM including multi-robot and multi-session SLAM frameworks. Retrieving previously seen intersections can also be advantageous from the point of view of large-scale outdoor topological mapping frameworks. Also, intersection recognition closely follows intersection detection [1], wherein the immediate question to answer upon detecting an intersection is if the detected intersection is the same as one seen previously. Intersection recognition from disparate video streams or image sequences is an extremely challenging problem that stems from the large variations in viewpoint, weather and appearance accross traversals. Complexity also emanates from varying levels of traffic and chaos at a junction, as well as due to changing levels of occlusion, illumination between two video sequences of a specific junction. Lack of annotated datasets on intersections also poses a challenge.

Within the robotics community, loop detection methods have used diverse image recognition and retrieval techniques [2], [3], [4] that have not attempted to detect loops from very diverse viewpoints, though they perform admirably in the presence of weather changes [5] or under the duress of varying traffic [6]. In the vision community, CNN [7] features have been used to efficiently compare scenes or structures which are viewed from varying distances, which give a zoom-in vis-a-vis zoom-out effect with minor changes in viewing angles.

Intersections being critical points in road based navigation, are likely to be points of crossover between trajectories of multiple agents/cars. Owing to their four-way structure, imagery captured in different traversals of junctions depicts very different views and alignment of common landmarks unlike what is observed in place recognition datasets(e.g., [8]). Therefore we believe that the problem needs attention and treatment of its own.

We take videos or image sequneces as the input to perform the recogntion task. The temporal information present inherently in such ordered image sequences leverages neighborhood information between frames, establishing relationships among objects in the scene. This also helps to mitigate the problem of perceptual aliasing to some extent, a problem that often plagues image-based recognition/relocalization approaches.

We propose a novel stacked deep network ensemble architecture that combines state-of-the-art CNN, Bidirectional LSTM and Siamese style distance function learning for the task of view-invariant intersection recognition in videos. While the CNN component of the ensemble primarily deals with image-level feature representation, the bidirectional LSTM is the key recurrent network component that enables learning of temporal evolution of visual features in videos followed by the Siamese network-based distance metric learning for comparing two input video streams. Our proposed network is conceptualized in Figure 2.

We contribute in the following ways:

1) Firstly, we propose a novel and pertinent problem of recognizing intersections when approached from highly disparate viewpoints
2) We propose an original deep network ensemble. The proposed architecture can handle videos of varying length and compare videos capturing reverse trajectories. This kind of Siamese network with recurrent LSTM component has been largely unexplored in the video domain. Furthermore, we use the hidden state of LSTM cells, instead of the traditionally used output state, for video representation, which has largely been unexplored too.

* Authors contributed equally
† Affiliated with KCIS, Center of Visual Information Technology, International Institute of Information Technology, Hyderabad, India
‡ Affiliated with KCIS, Robotics Research Center, International Institute of Information Technology, Hyderabad, India

Fig. 1. An example sequence pair from Mapillary IV-B.2 dataset. First row show frames (arranged in the temporal order) from the two trajectories respectively where we highlight the overlapping structural content (green and cyan). Second row zooms in for overlapping structures in the sequences. Rightmost figure in the row shows trajectory of both the videos in an *intersection-map* which acts as a *top-view* of the intersection. We color encode the trajectories and overlapping content in the *intersection-map*. (Best seen in color)

3) We showcase the efficacy of the proposed architecture through competitive results on challenging sequences while showing decent improvements from the image based baselines. We collect and annotate synthetic yet highly realistic data from the GTA [9] gaming platform and actual real world data of Mapillary [10] for this purpose.

To the best of our knowledge, this is the first attempt in this direction using the ensemble of state-of-the-art CNN, bidirectional LSTM and Siamese network architecture for recognizing intersection in input video pairs.

Figure 1 shows a sample sequence pair from Mapillary dataset (see Section IV-B.2) where both trajectories pass through same intersection. The pair of trajectories are depicted using two sequences of frames. We zoom into regions from both the trajectories to show corresponding common structures. We also show an *intersection-map* which represent the *top-view* of the trajectories on the intersection. The trajectories are perpendicular and hence minimal view overlap exists between them. In such cases, temporal information from a sequence of frames captures and reinforces the relationship between observed intersection landmarks during the traversal.

II. Literature Review

In existing literature around robotic perception, there are limited efforts towards intersection detection and recognition. Some of these use sensors other than cameras such as laser and virtual cylindrical scanners [11], [12].

Recently, a framework for detecting intersections from videos was proposed in [1]. They used LSTM based architecture to observe the relative change in outdoor features to detect intersection. Nevertheless, their work did not focus on the recognition task. On the other hand, visual loop closure detection techniques in the literature [13], [2], [3] mainly focused on image level comparison of scenes. These techniques try to find re-occurring scene during the driving session based on fast KD tree based comparison of image features. Although these methods achieve admirable accuracy for loop detection they cannot be improvised for view-invariant recognition over videos.

In recent years, gaming environments have been created and/or used for generating and annotating datasets. [14] used Grand Theft Auto V (GTA) gaming-engine to create large-scale pixel-accurate ground truth data for training semantic segmentation systems. SYNTHIA [15] is a virtual world environment which was created to automatically generate realistic synthetic images with pixel-level annotations for the task of semantic segmentation. [14] and [15] show the added improvement in the real-world setting by using synthetic-datasets for training deep-network models. CARLA [16] is another open-source simulator developed to facilitate learning and testing of Autonomous Driving algorithms. GTA environment is more visually realistic than SYNTHIA and CARLA environments.

Real-world datasets such as [17], [18] are designed with the idea of visual place recognition using images. These datasets are limited in the pathways covered and the number of intersections. Another real world street-level imagery platform [10] consist of images and image-sequences with variability in the pathways and intersections and thus enables sequence learning for intersection recognition. Concurrent works have used this platform to create Mapillary Vistas Dataset [19] which is a semantic segmentation dataset consisting of 25k images. We have used Mapillary dataset to showcase our result on real world scenarios.

Recently [20] and [21] have attempted visual localization under drastic viewpoint changes. [20] needs various modes of data including: semantic segmentations, RGBD images and camera parameters. Its a computationally expensive pipeline focused on relocalization, involving 3D voxel grid constructions of the entire previous traversal sequence and of the query sequence, followed by a matching and pose estimation procedure. Due to key differences in problem context and usage of multimodal inputs, we omit a quantitaive comparison with [20]. [21] proposes graph construction from semantically segmented input data with graph matching using random walk descriptors. We implement and compare

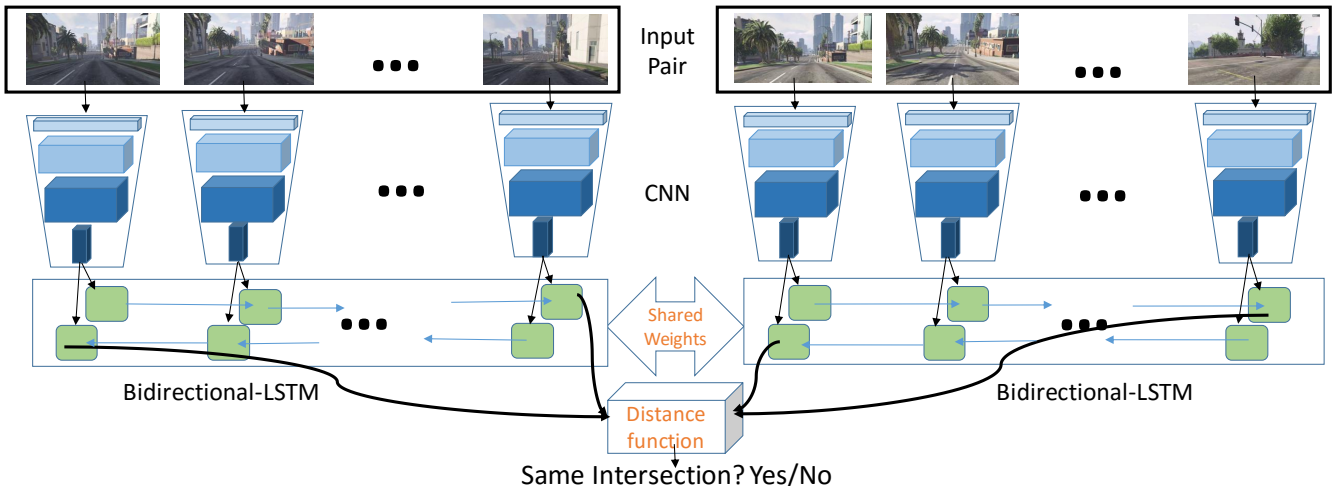

Fig. 2. Proposed model is shown with video pairs as input and binary-classification as the output. Features from pretrained CNN network are fed into bidirectional LSTM with shared-weights as shown with hidden units in green (unfolded in time).

against this method in a limited setup in our baseline experiments in Section IV-A.

Specifically there has been no prior effort towards recognizing intersections when approached from different road segments that constitute the intersection.

III. Method

In order to achieve view-invariant intersection recognition from video sequences, our overall approach follows the recent successes of deep network models in learning useful representations of visual data [22]. Our overall strategy comprises of three specific sub-objectives: (i) capture representations of individual video frames that can help identify the intersection in a view-invariant manner; (ii) leverage the temporal correlations between video frames to better capture the unique identity of an intersection across different views; and (iii) use an appropriate distance metric to compare the thus learned spatiotemporal representations of intersection video sequences. We use a stacked deep network ensemble architecture to realize the above strategy.

Our stacked ensemble consists of a state-of-the-art Convolutional Neural Network (CNN) to learn video frame representations, a Bidirectional Long Short-Term Memory (LSTM) Network (a variant of Recurrent Neural Networks, that addresses the vanishing gradient problem) that captures temporal correlations across video frames, and a Siamese Network that learns a suitable distance metric that can uniquely identify an intersection from these video sequences across views. Figure 2 summarizes the proposed ensemble strategy.

A. Convolutional Neural Networks (CNNs)

CNN based models have shown tremendous success in a variety of tasks like [23], [24], etc. and it has been shown that representations learned by the CNN can be used as off-the-shelf features for other tasks [25], [26].

From a plethora of CNN pretrained models, we choose the model most relevant to our tasks which provide invariance in lighting and pose. In particular, we employ AmosNet [27] as our CNN architecture. AmosNet is a variation of CaffeNet [23] trained for Visual Places Recognition under varying weather conditions. We use pooled outputs from last convolution layer of AmosNet as the CNN features.

B. Bidirectional LSTM

Long Short-Term Memory Network (LSTM) [28], is a variant of Recurrent Neural Network (RNN) with the ability to capture long-term dependencies [29]. It has a hidden representation which is updated at each time step, and also consists of a memory state which holds the relevant information at each time step. The information entering and exiting out of cell states is controlled by gating mechanisms (sigmoid, and tanh functions).

Bidirectional LSTMs [30] incorporate future and past context by running the inverse of the input through a separate LSTM. The proposed model employed Bidirectional LSTMs capture the temporal correlations between video frames. This enables our model to identify the intersection uniquely whether a vehicle approaches it in the forward or opposite direction. Instead of using averaged/fused output of unfolded LSTM units or output from last unit we use the hidden-representation from the last unfolded time step as a feature vector for the videos as shown in Figure 2.

C. Siamese Network

The third component of our deep network ensemble is a Siamese network, which is used to learn data-driven distance metrics. A Siamese Network [31] consists of two independent networks with same architecture and shared weights. Both these networks are merged to learn a distance metric, d, between the two inputs provided to the respective networks.

During training, Siamese networks work on triplets ($\mathbf{x}_1$, $\mathbf{x}_2$, y) where y is the ground truth similarity between $\mathbf{x}_1$ and

$\mathbf{x}_2$, i.e., $y=1$ if the videos denote the same intersection, else $y=0$. The networks' weights are then updated by minimizing the loss function described below.

Contrastive Loss: The total loss function over a dataset $X=\{(\mathbf{x}_1^i,\mathbf{x}_2^i,y^i),\forall i=1,\cdots,n\}$ is given by:

$$L_{\mathbf{W}}(X)=\sum_{i=1}^{n}L_{\mathbf{W}}^{i}((\phi(\mathbf{x}_1^i),\phi(\mathbf{x}_2^i)),y^i) \tag{1}$$

where $\mathbf{W}$ are the weights/parameters of the network, $\phi(\mathbf{x})$ denotes the output of the last layer of the shared network architecture for each individual input $\mathbf{x}$, and $L_{\mathbf{W}}^{i}$ is

$$L_{\mathbf{W}}^{i}=y^iL_{pos}(\phi(\mathbf{x}_1^i),\phi(\mathbf{x}_2^i))+(1-y^i)L_{neg}(\phi(\mathbf{x}_1^i),\phi(\mathbf{x}_2^i)) \tag{2}$$

where

$$L_{pos}(\phi(\mathbf{x}_1^i),\phi(\mathbf{x}_2^i))=d(\phi(\mathbf{x}_1),\phi(\mathbf{x}_2))^2 \tag{3}$$

$$L_{neg}(\phi(\mathbf{x}_1^i),\phi(\mathbf{x}_2^i))=\max(1-d(\phi(\mathbf{x}_1),\phi(\mathbf{x}_2)),0)^2 \tag{4}$$

At test time, the learned distance function is used to predict the similarity in the videos using the expression in Eq. 5.

$$\hat{y}=\begin{cases}0: d(\phi(\mathbf{x}_1),\phi(\mathbf{x}_2))>\theta\\ 1: d(\phi(\mathbf{x}_1),\phi(\mathbf{x}_2))\leq\theta\end{cases} \tag{5}$$

We set the value of θ to be 0.5 in our experiments. Since the number of possible negative pairs can be very high as compared to positive pairs, we scale the loss function for each positive by a constant factor (>1), given by the ratio of negative pairs to positive pairs in the dataset.

D. Training and Network Parameters

Although our model can be trained in an end to end manner we train it in a greedy way due to negligible effect on performance as compared to end-to-end training. We use pretrained weights from previous state-of-the-art CNN models which provide a meaningful weight-initialization to our model and reduces training time. We use the contrastive loss described in Eq. 1 to train the Bidirectional LSTM, as shown in Figure 2. The proposed model is trained using the ADAM optimizer which is a variant of Stochastic Gradient Descent (SGD).

In our initial experiments, we varied the number of Bidirectional LSTM-layers (upto 3), but the performance gain was negligible. Similarly, we experimented with different values for hidden-unit dimensions in the LSTM cell. Empirical results found 250 to be the best performing after which the performance saturates. Thus, we fix the number of Bidirectional layer to one and hidden layer dimension to be 250 in all further experiments.

IV. Experiments & Results

In this section, we first define the baselines used for comparison with our proposed method. Next, we proceed to provide description about synthetically generated and real datasets for various experiments. Subsequently we describe the various experimental scenarios which is followed by the explanation of the quantitative and qualitative results.

A. Baselines / Evaluation Settings

In this subsection, we discuss two different baselines for comparisons with our proposed method.

Siamese-CNN: We define a deep learning approach void of extrinsic temporal modeling using Siamese Network on CNN. Prior work on Siamese-CNN have included application for face-verification [32], one shot recognition [33] etc. First, we train a Siamese-CNN using positive and negative pairs of images depicting same and different intersections respectively. Then, given two videos ($\mathbf{x}_1$, $\mathbf{x}_2$) comprising of n and m frames respectively, we use learned Siamese-CNN for prediction (using Eq. 5) for all possible frame comparisons (nm scores) and combine these predictions into a single score by computing their sum. We then learn a threshold maximizing the accuracy. During testing we then use this threshold in Eq. 5 to predict similarity in videos. For fair comparison the Siamese-CNN is initialized with same pretrained weights.

X-View: For the purpose of comparison with recent works on drastic-viewpoint changes we focus on [21]. Since [21] uses multiple data-modalities, for a fairer comparison, we limit the data-modalities to those strictly necessary for the model, using robot odometry (GPS coordinates and heading changes) and semantic-segmentation on RGB images. We ignore the backend needed in [21] as we do not focus on localization task, instead replacing it with a match pruning procedure. Each node in the query graph votes for k frames in the memory graph, based on matches with component nodes of these frames. Windows of frames in the memory sequence covering a threshold of votes from the distinct nodes/frames from the query graph, are taken as candidate matches corresponding to the query sequence.

We found this method to be prone to *semantic* perceptual aliasing, as semantic blobs of the same class are non-discriminative as their neighboring nodes' relations are their only descriptors. Thus, a direct comparison of [21] in the same context as ours (one shot recognition, instead of relocalization within a previous traversal sequence) wouldn't be meaningful.

B. Dataset Description

Dataset Collection: Each video/sequence consist of a series of frames collected around junction points. Each junction can be traversed in multiple pathways. We refer to the set of all such possible traversals as *trajectories*.

1) GTA: Here, we collected videos in two different scenarios. Firstly, we choose a set of intersections in the game-environment and then sample them in a *dense* manner, i.e., each possible trajectory in the junction is traversed. Secondly, we traversed a car from one point to another via means of an AI-car driving mod [34]. From such traversals, we break the captured video into small chunks involving intersections and non-intersections. We discard video chunks without intersections.

In the first scenario, we selected 27 unique junctions where 9 of them had arbitrary lighting conditions and the rest were in normal daylight condition. In the second scenario,

Fig. 3. Data Visualization: Random snapshots from GTA environment [9] (Row 1) and Mapillary [10] (Row 2). Images show different weather and day/night conditions as well as various outdoor scenes for urban scenarios. The game-environment snapshots look visibly similar to that of real world images and have significantly higher variations in weather and lighting. (Best seen in color)

TABLE I

NUMBER OF VIDEO PAIRS IN TRAINING, TESTING AND THE VALIDATION SET FOR THE DIFFERENT EXPERIMENTAL SCENARIOS

Dataset	Lighting Setting	Trajectory Relation	Training		Validation		Testing	
			Positive	Negative	Positive	Negative	Positive	Negative
GTA	day and night	All combinations	1509	1440	459	612	106	104
		Overlap in view and trajectory	640	1440	168	212	62	104
	day	All combinations	864	1166	300	653	72	82
Mapillary	day	Overlap in view and trajectory	3080	3328	421	428	400	400
		All combinations	6409	6976	350	318	425	420

we captured 12 traversals in two arbitrary different lighting variations.

2) Mapillary: We download images from [10], which is a community-led service with more than 200 million street-level images. These images are captured in various modes including walking, riding (either a bike or car), panorama and photo-spheres. We use mapillary's API to download images and construct *trajectories*. We first query for all the images in a bounding box defined by the longitudes and latitudes. For every image in the bounding box, the API provides a tuple consisting of image-key, latitude, longitude, orientation and the (video-)sequence it belongs to along with other metadata information. These images can be then downloaded using the image-key. Using two orthogonally overlapping sequences we get a location in the map which is used to identify the intersection. Subsequently, this is used to download all sequences (a set of continuous images) passing through that intersection. We mined around $300,000$ images which consisted of around 1700 usable sequences from around 500 junctions.

Figure 3 shows random snapshots captured from GTA environment (Row 1) and Mapillary (Row 2). One can infer from the figure that the snapshots from GTA environment are visually realistic as well as similar to real-world data from Mapillary and contain the key challenges associated with real environments (as listed in Section I). However, the GTA environment offers more complexity in terms of weather and lighting variations as compared to Mapillary.

Dataset Annotation: From the extracted dataset, we first annotate positive and negative video pairs. Positive (videos involving the same intersection) pairs are mined from the above datasets exhaustively, i.e., we select all possible positive pairs. Negative pairs generated from the datasets can be very high as any two trajectories from different junction are treated as a negative pair. Hence we limit the number of negative pairs by randomly fixing a subset. We keep the test, train and validation sets mutually exclusive by sampling them from different junction and/or non-overlapping traversals.

C. Experimental Setup

For a pair of videos involving the same junction there exists a trajectory-relation between them, which can be categorized in terms of overlap in view and/or trajectory. Overlap in trajectory refers to the proximal relation in the two trajectories while overlap in view refers to trajectories, viewing the larger part of the same area. For example trajectories moving in opposite direction can be proximal in distance but can have minimal view overlap.

For the purpose of our experiments we categorize the trajectory-relations into two primary setups: *Overlap in view and trajectory* and *All combinations*. The former refers to parallel and overlapping trajectory-relations while the latter refers to all the possible trajectory-relations. Similarly, we categorize the lighting setting into two categories: *day* vs *day and night*. The names are self-explanatory.

We test the robustness and generalizing capability of our model under various trajectory-relation and lighting (day/night) settings. The details of the experiment-wise distribution of samples can be found in Table I.

Evaluation Metric: We report percentage accuracy and F1 as the metric in our experiments. Accuracy is defined as the ratio of the number of correctly classified samples (both positive and negative pairs combined) to the total predictions made. F1 is defined as the harmonic mean of precision and recall. We note that while accuracy focuses on both positive and negative predictions, F1 is focused toward positive predictions (as precision and recall focuses on true-positives).

TABLE II

ACCURACY, PRECISION, RECALL AND F1 SCORE (F1) OF OUR METHOD ON DIFFERENT DATASETS FOR THE DIFFERENT EXPERIMENTAL SCENARIOS

Metrics ↓	Dataset →	GTA			Mapillary	
	Lighting Setting →	day and night		day	day	
	Trajectory Relation→	All combinations	Overlap in View and Trajectory	All combinations	All combinations	Overlap in View and Trajectory
Accuracy	Our Method	**70.95**	**78.2**	**76.72**	**72.1**	**81.0**
	Siamese-CNN	65.3	65.3	68.0	63.0	71.6
F1	Our Method	**62.3**	**60.0**	**72.7**	**61.9**	**82.4**
	Siamese-CNN	47.9	52.8	51.1	59.0	73.8

D. Quantitative Results

Table II shows performance on various metrics on GTA and Mapillary dataset under varying scenarios using our proposed model and baseline Siamese-CNN. We show different comparisons based on lighting setting, trajectory relation and baseline approach.

GTA: We observe that our proposed method performs better than the Siamese-CNN baseline on both Accuracy and F1. Additionally, among different models in the proposed approach (for different lighting conditions) we observe that the model improves on accuracy (by 5.76%) and F1 (by 10.4%) in *day* scenario as compared to *day and night* scenario. Similar trend of values is also observed in the Siamese-CNN baseline. This trend of results can be attributed to the reduced complexity in lighting variations.

Interestingly, under the *day and night* condition, the model performs better at accuracy (7.25% increase) when video-pairs have an *Overlap in view and trajectory* as compared to *All combinations*. However we observed an inverse trend for F1 measure. This anomalous behavior can be explained based on the performance on negative samples in this scenario. In this case we found that the true negative rates performance for *Overlap in view and trajectory* is 83.0% as compared to *All combinations* at 52.8% (30% increase). Nevertheless, the performance of proposed model is still significantly higher than baseline method.

Mapillary: Similar to GTA, we observe that our proposed method performs better than the Siamese-CNN baseline on both accuracy and F1. Among models in our proposed method, the performance is better on all metrics (accuracy: 8.9%increase, F1:20.5%increase) when the trajectory-relations are limited to *Overlap in view and trajectory* as compared to *All combinations*. Similar change in values can be observed in the Siamese-CNN baseline.

GTA vs Mapillary (for proposed method): In the common scenario of *day* and *All combinations*, we notice that the performance on GTA is better than Mapillary for accuracy (4.62% increase) and F1 (10.8% increase). This can be explained due to complexity of data annotation in real-world where visually dis-similar videos can be annotated as same based on their GPS coordinates.

Table III shows experimental results of X-view on Mapillary in relocalisation scenarios. We used 12 relevant and reliably segmented semantic categories. Localisation Threshold(maximum distance from Ground Truth for a match) and Inlier Frames%(% of dictinct valid frames voting for a window of frames in the memory sequence) were varied to obtain the best F1 scores. We observe that F1 scores obtained are always lower than F1 scores for Mapillary data in Table II. The method had high recall but low precision across multiple thresholds. This can partly be attributed to the fact that [21] requires 15 or more overlapping frames for reliable localization whereas in case of orthogonal trajectories in Mapillary only 5-7 frames have overlapping structures.

TABLE III

X-VIEW RESULTS ON MAPILLARY: F1-SCORES AT DIFFERENT EXPERIMENTAL THRESHOLD (QUERY SEQUENCE LENGTH: 15)

Localization Threshold	**Inlier Frames %**	**F1**
20m	70	51.5
40m	70	50.9
40m	80	47.7
60m	70	43.7
60m	80	46.7

E. Qualitative Results

Figure 4 shows 3 positive samples from Mapillary that are successfully predicted by the network. Each sample depict a unique trajectory-relation and view-overlap as shown by the *intersection-map* (bottom-most row). We color encode (green, cyan, red, brown) the visually perceptible common structures (using ellipses) in the trajectories. We follow the same encoding while plotting the respective *intersection-map* in the last row. In video pair 1 and video pair 3, we observe that the overlapping part of trajectories are in the opposite direction but our network managed to exploit the common structures seen from different views. In video pair 2, though the partially overlapping trajectories are in the same direction, we can observe that *shadows* pose a challenge that is successfully handled by the network. This indicate the ability of proposed network to generalize on varying lighting conditions. Moreover, we also observe occlusion in a few trajectories (video pair 3 trajectory 2) due to vehicles.

Figure 5 depicts 2 positive samples from GTA that are successfully predicted by the network. For example: video-pair 1, capture the different view of the same structure in rainy weather. Similarly, in video-pair 2 the model is able to predict similar videos in presence of view and lighting variations. Due to space constraints we only show 6 aligned contiguous frames in these video-pairs, where we manually crop the relevant contiguous frames for illustration purpose.

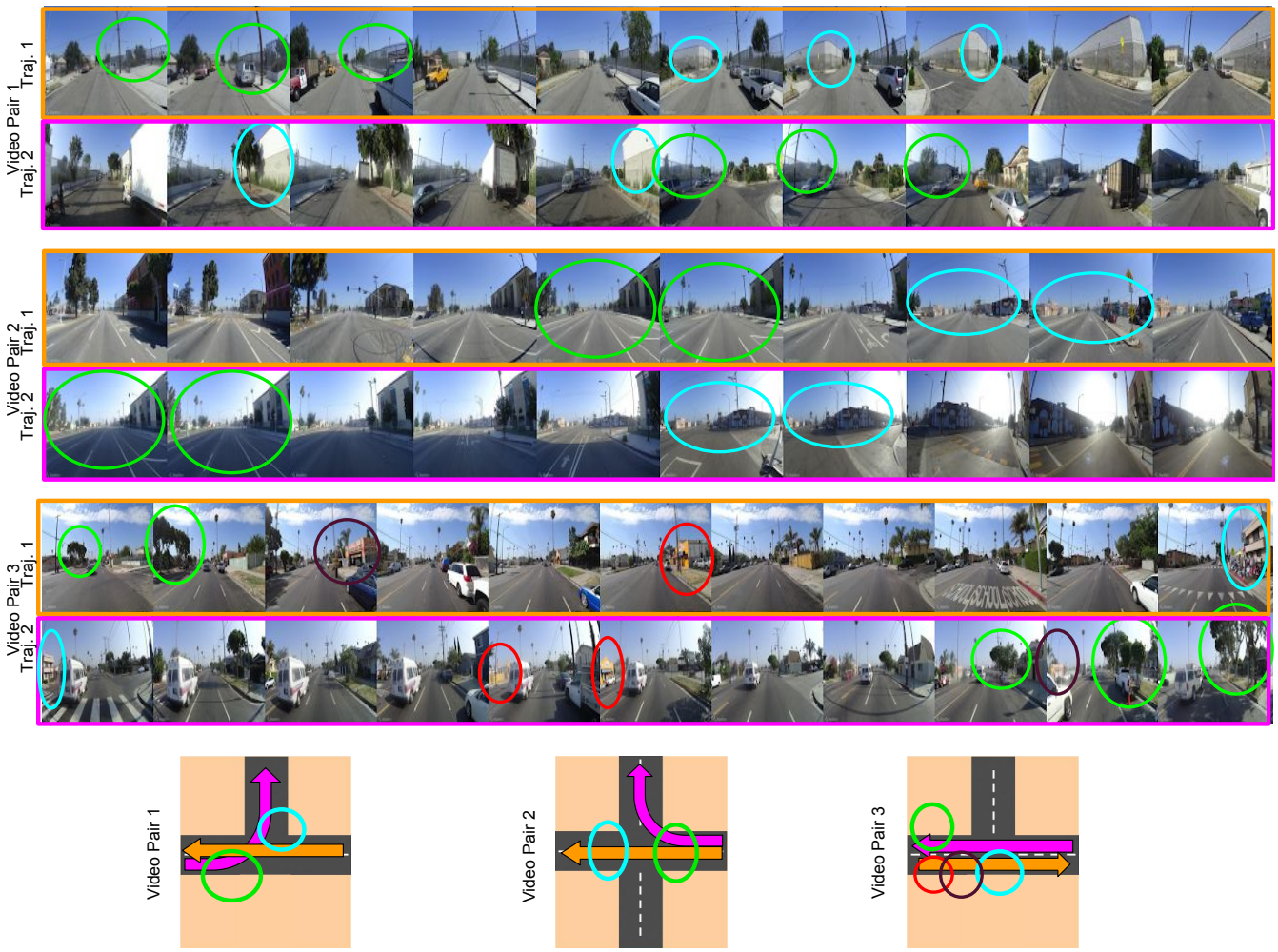

Fig. 4. Three correctly classified samples from Mapillary. Each sample depicts unique trajectory-relation and view-overlap between trajectories as shown in the *intersection-map* (bottom-most row) which represents the *top-view* of the intersection. We color encode the visually perceptible common structures (cyan, green, red, brown) in the Video pairs. Video pair 2 and Video pair 3 show examples where trajectory direction and view overlap. Video pair 1 depict cases when the overlap in trajectories and/or the view is minimal. (Best seen in color)

Fig. 5. Two correctly classified samples from GTA. Video pair 2 show examples where trajectory direction and view overlap. Video pair 1 depict case when the overlap in trajectories and/or the view is minimal. (Best seen in color)

Figure 6 shows some incorrect predictions made by the network. Video pair 1 consist of trajectories from different intersections (negative sample) but is predicted as the same intersection. We believe this may be due to lack of unique structures in the scene as most of it has trees. Additionally, the variance in lighting conditions (as the network was trained on *day* conditions in Mapillary) can also pose a challenge if not trained on other conditions. In contrast, video pair 2 are trajectories from the same-intersection but are predicted as different intersection. In this video pair, we observe a complete lack of visual features in the overlapping part of trajectories. Overlapping structures are completely occluded in the first trajectory due to the presence of *truck*. Uing purely visual content for all these failure cases, even humans might easily fail without extra context (ground truth labels were generated using GPS coordinates).

V. CONCLUSIONS

This paper proposed a novel stacked deep network ensemble architecture that combines state-of-the-art CNN, bidirectional LSTM and Siamese style distance function learning for

Fig. 6. Failure Cases: Two wrong predictions by the network. Video pair 1 consist of trajectories from different intersections but is predicted as the same intersection. In contrast, video pair 2 consist of trajectories from the same-intersection but are predicted as different intersection.

the task of view-invariant intersection recognition in videos. The proposed architecture enables recognizing the same intersection across two videos of variable length having large view variations, inverted trajectory, lightning and weather variations. We have collected annotated data (more than 2000 videos) from GTA [9] and Mapillary [10] and have computed results on this data with varying parameter choices and have reported competitive results.

REFERENCES

[1] D. Bhatt, D. Sodhi, A. Pal, V. Balasubramanian, and K. M. Krishna, "Have I reached the intersection: A deep learning-based approach for intersection detection from monocular cameras," in *IROS*, 2017.

[2] A. Glover, W. Maddern, M. Warren, S. Reid, M. Milford, and G. Wyeth, "OpenFABMAP: An open source toolbox for appearance-based loop closure detection," in *ICRA*, 2012.

[3] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *TRO*, 2012.

[4] S. Achar, C. V. Jawahar, and K. M. Krishna, "Large scale visual localization in urban environments," in *ICRA*, 2011.

[5] C. Linegar, W. Churchill, and P. Newman, "Made to measure: Bespoke landmarks for 24-hour, all-weather localisation with a camera," in *ICRA*, 2016.

[6] A. A. Hafez, M. Singh, K. M. Krishna, and C. V. Jawahar, "Visual localization in highly crowded urban environments," in *IROS*, 2013.

[7] N. Sunderhauf, S. Shirazi, A. Jacobson, F. Dayoub, E. Pepperell, B. Upcroft, and M. Milford, "Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free," in *RSS*, 2015.

[8] M. J. Milford and G. F. Wyeth, "Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights," in *ICRA*, May 2012, pp. 1643–1649.

[9] "Grand Theft Auto V," https://en.wikipedia.org/wiki/Development_of_Grand_Theft_Auto_V.

[10] "Mapillary," https://www.mapillary.com/app.

[11] Q. Zhu, Q. Mao, L. Chen, M. Li, and Q. Li, "Veloregistration based intersection detection for autonomous driving in challenging urban scenarios," in *ITSC*, 2012.

[12] Q. Li, L. Chen, Q. Zhu, M. Li, Q. Zhang, and S. S. Ge, "Intersection detection and recognition for autonomous urban driving using a virtual cylindrical scanner," *IET Intelligent Transport Systems*, 2013.

[13] A. Angeli, D. Filliat, S. Doncieux, and J. Meyer, "Fast and incremental method for loop-closure detection using bags of visual words," *TRO*, 2008.

[14] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *ECCV*, 2016.

[15] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. Lopez, "The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *CVPR*, 2016.

[16] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *CoRL*, 2017.

[17] M. J. Milford and G. F. Wyeth, "Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights," in *ICRA*, 2012.

[18] "Gardens Point Dataset," https://wiki.qut.edu.au/display/cyphy/Day+and+Night+with+Lateral+Pose+Change+Datasets.

[19] G. Neuhold, T. Ollmann, S. R. Bulò, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in *ICCV*, 2017.

[20] J. L. Schönberger, M. Pollefeys, A. Geiger, and T. Sattler, "Semantic visual localization," *arXiv preprint arXiv:1712.05773*, 2017.

[21] A. Gawel, C. Del Don, R. Siegwart, J. Nieto, and C. Cadena, "X-view: Graph-based semantic multi-view localization," *arXiv preprint arXiv:1709.09905*, 2017.

[22] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *TPAMI*, 2013.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[24] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *arXiv preprint arXiv:1511.00561*, 2015.

[25] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN Features off-the-shelf: an astounding baseline for recognition," in *CVPR workshops*, 2014.

[26] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.

[27] Z. Chen, A. Jacobson, N. Sunderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford, "Deep learning features at scale for visual place recognition," *arXiv preprint arXiv:1701.05105*, 2017.

[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, 1997.

[29] A. Karpathy, J. Johnson, and L. Fei-Fei, "Visualizing and understanding recurrent networks," *arXiv preprint arXiv:1506.02078*, 2015.

[30] M. Schuster, K. K. Paliwal, and A. General, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, 1997.

[31] W. Yih, K. Toutanova, J. C. Platt, and C. Meek, "Learning discriminative projections for text similarity measures," in *CoNLL*, 2011.

[32] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf.

[33] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML-W*, 2015.

[34] "Grand Theft Auto V Auto-drive Mod," https://www.gta5-mods.com/scripts/vautodrive.