Synthesizing Attributes with Unreal Engine for Fine-grained Activity Analysis

Tae Soo Kim Mike Peven Weichao Qiu
Alan Yuille Gregory D. Hager
Johns Hopkins University
{tkim60,mpeven1,wqiu7,alan.yuille,hager}@jhu.edu

Abstract

We examine the problem of activity recognition in video using simulated data for training. In contrast to the expensive task of obtaining accurate labels from real data, synthetic data creation is not only fast and scalable, but provides ground-truth labels for more than just the activities of interest, including segmentation masks, 3D object keypoints, and more. We aim to successfully transfer a model trained on synthetic data to work on video in the real world.

In this work, we provide a method of transferring from synthetic to real at intermediate representations of a video. We wish to perform activity recognition from the low-dimensional latent representation of a scene as a collection of visual attributes. As the ground-truth data does not exist in the ActEV dataset for attributes of interest, specifically orientation of cars in the ground-plane with respect to the camera, we synthesize this data. We show how we can successfully transfer a car orientation classifier, and use its predictions in our defined set of visual attributes to classify actions in video.

1. Introduction

Photo-realistic simulation of the real world enables the generation of large scale visual datasets with rich set of annotations. Technical advances coupled with wider adoption of synthetic datasets facilitated advances in applications including self-driving cars [2], object parsing [4] and optical flow estimation [1]. However, how to properly harness the full potential of synthetic datasets for fine-grained activity analysis remains an open problem. In this work, we model a complex activity as a collection of simpler visual attributes. Mapping a video into a temporal stream of attributes provides a low-dimensional and interpretable representation from which to classify activities. Furthermore, it presents a natural way to use synthetic data to fill in the gaps of a dataset which lacks ground-truth labels for attributes of interest. In this work, we present our recent approach for synthesizing visual attributes of interest (car orientation) and how we apply the synthetically trained model for recognizing *Entering* activity of the ActEV dataset.

2. Methods

2.1. Synthetic Data Generation with Unreal Engine

Our synthetic data generation system is based on Unreal Engine (UE) including four modules.

The first module is the scene generation module. It contains 3D assets and scene generation code to layout 3D objects. Our scene generation code parameterizes the scene: the number of surrounding buildings, the trajectory of the car motion, the density of the parking lot, etc. The second is the motion capture module. Typical human motions such as running and walking can be found in the public UE Marketplace[1]. The UE Marketplace is a great source for finding assets for building entertainment games. However, the market does not have motion data for complex interactions such as opening the door and entering a car which are less relevant for game designers. So we built our own motion capture pipeline based on Noitom motion capture hardware. This enables us to map real human movements into the virtual space. The third is the domain randomization module. Domain randomization is crucial for training vision models that transfer to real imagery. Our system supports randomized lighting conditions, camera extrinsics and material control. These are generic building blocks that can be reused and shared for any applications which requires randomization. The last module is the data capture module for capturing rich ground truth data from virtual scenes, which includes 3D keypoints of car and human, object segmentation, part segmentation, depth, surface normal, the begin and end point of an activity in the timeline.

We provide an API to UE components in C++ and Blueprint (scripting language provided by UE) to easily account for new use cases. Furthermore, we provide a python interface which enables on-line learning settings such as reinforcement learning.

[1] https://www.unrealengine.com/marketplace/store

Figure 1. The first image (from left to right) is a synthetically generated sample from our UE4 system. Our synthesis engine provides detailed scene information such as actor positions and actor key points. The visual similarity between the first and the second sample illustrates the flexibility of our system effectively mimic the visual characteristics of the target dataset of interest. The third image is a sample from our synthetic car orientation dataset. The fourth image is a test sample from the WCVP [5] dataset. Best viewed in color.

2.2. Relative Geometry of Cars and Humans

For modeling fine-grained interactions of cars and humans, we identified that the capability to analyze relative geometry of cars and humans is critical. We leverage on our capability to predict car orientation (discussed in 3.1) given a car crop. First, we initialize a mask for car-human relative position with three labels, 'front', 'rear' and 'side', assuming that the vehicle is facing the 0° direction. Then, we rotate the mask using the orientation prediction, superimpose the human detection bounding box onto the mask and take the mode of the mask values within the bounding box as our final relative position prediction.

2.3. Activity Analysis with Visual Attributes

We model activities as a collection of predefined visual attributes. We classify that a certain activity has occurred when we observe all defined visual attributes in a given segment of video. In this work, we focus on one DIVA-V1 class, *Entering* (a vehicle). Given a human actor, H1, and a car actor, C1, we define *Entering* as a collection of the following attributes: {*Exists*(H1), *Exists*(C1), *SideOf*(C1,H1), *Disappears*(H1)}. We can compute *Exists* and *Disappears* attributes from detection results. *SideOf* attribute classifier is trained using synthetic data which is discussed in 3.1 and 3.2.

3. Experimental Setup and Results

3.1. Car Orientation

We finetune a resnet implementation in PyTorch for learning car orientation. This problem is posed as a 36-way classification task, with each label corresponding to a 10° bin of the car orientation in the ground-plane with respect to the camera. We generated 250K images with UE for training and used two datasets [3] and [5] for testing (as no ground-truth orientation exists in the ActEV). Results are shown in table 1.

3.2. Relative Geometry and Entering

To validate our approach for relative geometry prediction using car orientation model on the ActEV dataset, we use activity labels as a proxy for extracting relative geometry between cars and humans. We first labeled images from the *Open_Trunk* and *Closing_Trunk* classes as (human is) 'rear', and labled crops from *Entering* and *Exiting* as 'side'. Then, we manually filtered out crops where a human does not exist. We show our experimental result on classifying relative geometry in Table 3.2 where the baseline model is a resnet trained on synthetic data to directly predict 'side', 'front' and 'rear'. We synthesized 100K images where a human was randomly placed proximal to a car. We show that the geometry based model using orientation shows better quantitative results. The presented baseline system for P_miss analysis in Table 3.2 is our deep neural network based recognition model which is trained to classify a spatio-temporally localized video into all ActEV classes. Given a set of localized regions of interest in video, the network classifies segments in a sliding window fashion. A simple attribute based approach leads to significant improvements validating the potential of data synthesis in action analysis.

Table 1. Car orientation results

Dataset	Accuracy	Top-3 Accuracy (within 20°)
EPFL	24.19%	57.74%
WCVP	23.86%	57.58%

Table 2. The first row presents experimental results of the proposed relative geometry prediction model on ActEV dataset. The second row evaluates the approach proposed in Section 2.3.

	Baseline	Proposed
Relative Position Accuracy	52.6%	62.5%
Entering P_miss @ 1rfa	0.760	**0.295**

4. Conclusion

We presented our approach for synthesizing visual attributes using Unreal Engine for modeling fine-grained activities. We generated a synthetic dataset to learn car orientations which we then use to improve recognition performance on *Entering* activity of ActEV.

Acknowledgement

This work is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00345. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/IBC, or the U.S. Government.

References

[1] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision*, pages 611–625. Springer, 2012.

[2] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4340–4349, 2016.

[3] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich. Viewpoint-aware object detection and continuous pose estimation. *Image and Vision Computing*, 2012.

[4] C. Li, M. Z. Zia, Q.-H. Tran, X. Yu, G. D. Hager, and M. Chandraker. Deep supervision with intermediate concepts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018.

[5] M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 778–785. IEEE, 2009.