Synthetic Data as a Supplement for Training Deep Learning Models

Jatin Katyal

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Computer Science at

Concordia University

Montréal, Québec, Canada

May 2023

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Jatin Katyal**

Entitled: **Synthetic Data as a Supplement for Training Deep Learning Models**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Adam Krzyzak

_____ Examiner
Dr. Adam Krzyzak

_____ Examiner
Dr. Eugene Belilovsky

_____ Supervisor
Dr. Charalambos Poullis

Approved by _____
Dr. Lata Narayanan, Chair
Department of Computer Science and Software Engineering

_____ 2023

_____
Dr. Mourad Debbabi, Dean
Faculty of Engineering and Computer Science

Abstract

Synthetic Data as a Supplement for Training Deep Learning Models

Jatin Katyal

Training supervised machine learning models with suitable data can be challenging due to the expensive and time-consuming process of collection and annotation, particularly when publicly accessible datasets are not available. This work emphasizes the use of synthetic data to reduce the effort spent on data collection. The study is based on an analysis of three widely-used benchmark datasets and two synthetic datasets one of which was created for this specific task. The insights derived from the preliminary analysis identify the required characteristics of the synthetic data that can consistently lead to improvement in performance metric for the task. In this work, the roles of synthetic data as a supplement to real data is investigated. Precisely, the impact of synthetic data with varying proportions and similarities to real data on the performance of Multiple Object Tracking neural networks based on the Convolutional and Transformer architectures is analyzed. Experiments demonstrate the superiority of using a combination of simulated and real data, where the samples of synthetic data are many folds of the real, and the variance of low-level features is high but limited by the low variance of high-level features. The findings can be applied to other machine learning tasks and provided guidelines can help improve model performance.

Acknowledgments

Allow me to take this moment to thank everyone who helped make this journey possible.

I want to start by expressing my deep thanks to my supervisor, Dr. Charalambos Poullis, for welcoming me to his lab and supporting my significant growth during the master's programme. His perseverance and guidance were crucial to continually improve this research. I consider myself quite lucky to have had him as my academic advisor.

The work reported in this thesis wouldn't have been possible without the financial support from NSERC, I am grateful to them.

Last but not least, I want to thank my mother, brother and friends for their unwavering love and support, which helped me through the difficult times by allowing us to enjoy little moments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The ever-growing internet opens the frontiers for data collection through keyword searches and scraping websites. These methods are often constrained by the absence of labels or the substantial effort required in labelling. Techniques such as crowd-sourcing and active learning mitigate these constraints but they come with their own drawbacks. Crowd-sourcing platforms are limited by uncertainty due to human diligence, the latency between setting up the task on a marketplace & receiving results, and the compensation of the workers involved [1]. The benefits of Active Learning are restricted by the scalability of large datasets and the involvement of a human oracle [2]. In addition, the use of copyrighted materials must be approached with the appropriate ethical and legal considerations, and sensitive personal information must be handled with care, requiring approval from appropriate ethics committees or other governing bodies. International incidents can raise geopolitical tensions over data collection as in the case of a high-altitude balloon [3] alleged for surveillance and meteorological purposes by two conflicting nations. Thus, gathering high-quality labelled samples not only requires substantial time and resources but, can have serious consequences if the collection doesn't follow through the proper channel. If the collection of data was the sole purpose of this aircraft, proprietors could have benefitted from the multiple organizations that make the information accessible to the general public.

The feasibility of leveraging publically accessible datasets seem more practical given the economics and resource expenditures associated with the collection of a new dataset. It also eliminates the aforementioned challenges. In practice, however, straightforward application of such datasets

may not always be feasible or effective due to a variety of potential challenges or constraints.

The challenges include but are not limited to the following. Data leakage and duplicity issues in datasets can reduce their eminence for training deep learning models [4]. Duplicates in a generative setting can output results the same as training set samples which could be an imitation of copyrighted material [5]. Outdated datasets that may not accurately reflect current or emerging scenarios or environments [6]. This can be due to a lack of samples with variations (weather, illumination, pose etc.) leading to overfitting and low accuracy in congested scenes [7]. It is also hard to address the class imbalance issue on arising tasks using public datasets as the problem continues to evolve [8]. Even in the existing problem statements, the increasing complexity over the years and the rarity of specific data in nature are becoming challenging [9]. Also, it is cumbersome to control the distribution of variables in large-scale datasets and negligence can result in biases that are fatal in many applications [9]. At times, the large-scale datasets are straightaway restricted to the general public [10]. Discussed in [11], are the privacy concerns and the privacy-protecting actions taken by the authorities over the world. These limitations of public datasets reduce their utility for training deep learning models.

One potential solution to these challenges is utilizing synthetic data in addition to available limited real-world datasets. The difficulty of collecting a large number of training samples for deep learning tasks is overcome by simulating diverse scenarios that are automatically labelled without incurring additional costs [11, 12]. It is demonstrated in [9, 13], that the challenges of bias and privacy can be mitigated with synthetic data. Troubles with overfitting, class imbalance and data rarity have also been benefited by the use of synthetic data [7, 8, 9]. Thus by supplementing real-world data with synthetic data, the limitations inherent to traditional data sources can be overcome.

The adoption of synthetic data has already been investigated in recent literature, in [7, 8, 14] focus is on domain adaptation while in [9, 10, 12, 13] emphasis is on pre-training with synthetic datasets. Although both of these methods have their own merits, in this work another approach is proposed which is both simple and straightforward. It involves the direct utilization of synthetic data without the need for additional domain adaptation or pertaining steps. This technique is justified by viewing the domain adaptation or pre-training steps as potentially costly and extra when dealing with an already challenging task.

This study investigates how artificial data affects machine learning model performance. As a first step, a comparison between the synthetic and actual sequences is drawn using the Frechet Inception Distance (FID)[15]. In accordance with observed patterns, clusters for low-level and high-level features are formed which raises the question of whether and how these clusters affect the performance of the models. Following that, the impact of different concentrations of real-synthetic data in the training set and sequences belonging to different clusters is assessed. The positive role of synthetic data on three benchmark datasets is demonstrated with a convolutional and a transformer architecture in a Multi Object Tracking task. Building upon the findings, a set of guidelines for using synthetic datasets along with actual data are also provided.

The following contributions are presented:

- **Investigation of the efficacy of integrating synthetic data with real data for improving the performance of Multiple Object Tracking.** Studies have been conducted to blend in synthetic data generated through advancements in graphics engines for vision-related tasks like classification, detection, segmentation, and re-identification. Most of these works utilized a domain adaptation or pre-training approach. In this work, a direct application without such additional techniques on Multiple Object Tracking tasks is examined.

- **Comprehensive analysis of experimental results and insights on the synergistic effects of using synthetic data.** In the presented work, the symbiotic effect of synthetic and real data on training deep learning models is studied. Is all synthetic and real data equal? Does the use of synthetic data induce a positive change in the performance of tracking measures? If there's a change, is it influenced by the amount of synthetic data present in the training set? Does the impact differ for different data?

- **General recommendations for generating and incorporating synthetic data to enhance model performance.** After investigating the efficacy of the synergistic effects, this work discusses on how to achieve similar results by adopting a general set of guidelines. The discussion is on the lines of the amount of synthetic data that should be used, the factors that influence the increase in performance and the factors that limit the performance increase. The discussion also includes the performance increase observed during the experimentation

presented in this work.

1.1 Notes on Contribution

This thesis presents the manuscript under preparation ”Synthetic Data as a Supplement for Training Deep Learning Models”, Jatin Katyal, Charalambos Poullis. Jatin Katyal’s contributions to this work include background research, generation of a synthetic dataset, preliminary analysis among the real and synthetic datasets, experimentation and writing shown in Chapter 3. Dr. Charalambos Poullis’s major contribution includes supervision, guidance and manuscript review.

1.2 Thesis Organisation

The thesis is organized as follows: Chapter 2 accounts for a brief survey of previous works in regard to Deep Learning, Synthetic Data and Multiple Object Tracking. Chapter 3 presents the analysis of features of synthetic and real data and demonstrates the improvement in tracking metrics when using a combination of both under a Multiple Object Tracking scenario. experimental results and conclusions. Chapter 4 shows the additional results using the technique discussed in Chapter 3. Finally, Chapter 5 concludes this work with a scope for future works.

Chapter 2

Background

Abstract

This chapter dives into a brief history of neural network-based machine learning, some of the methodologies previously employed to create synthetic data for computer vision tasks and different object-tracking techniques. By exploring these diverse approaches, we aim to provide a comprehensive overview of the techniques in the field of synthetic data and object tracking using deep learning.

2.1 Deep Learning: A brief survey of neural network architectures

The review of the architectures is divided into three categories, early Artificial Neural Networks (ANNs), Traditional ML for computer vision and Deep Learning.

2.1.1 Early Neural networks

To describe how the biological brain makes decisions, [16] developed the MP-Neuron, which eventually served as the basis for all familiar neural network topologies. In [17] a classifier was modelled that could classify linearly separable classes by learning weights. In [18], LeNet was presented for handwritten digit recognition; this would subsequently serve as the inspiration for modern methods for resolving computer vision problems.

2.1.2 Modern Computer Vision

The modern era computer vision focussed on handcrafted features like SIFT[19], SURF[20], ORB [21] often coupled with traditional machine learning techniques like Support Vector Machines, K-nearest neighbours, Naive-Bayes, Decision Trees and Random Forests [22, 23]. In general, one or more such handcrafted features will be selected for extraction. These extracted features are then utilized to train a traditional Machine Learning algorithm for tasks such as image classification. The art of feature engineering got redundant as more complex Artificial Neural Network architectures replaced the conventional machine learning approaches and deep learning gained traction.

2.1.3 Deep Learning Era

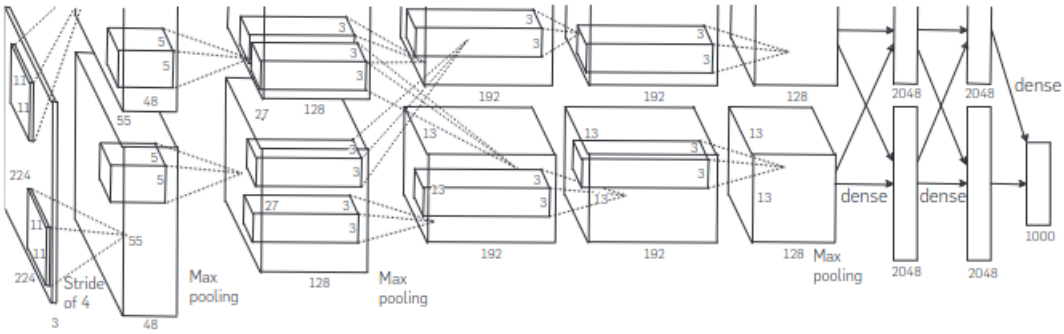
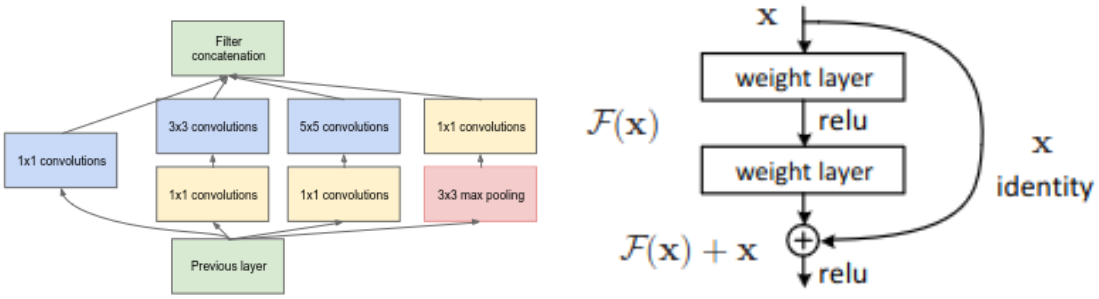


Figure 2.1: AlexNet architecture with five convolutional and three fully connected layers

The introduction of AlexNet in [24] revolutionized the field of computer vision by optimizing a convolutional model to GPU hardware for the task of image classification on a very large dataset. Since the dawn, there have been numerous developments in the architectural aspect. VGG-16 and VGG-19 architectures introduced in [25], focussed on increasing the depth of the model. Previously popular AlexNet had eight layers but these two architectures had Sixteen and Nineteen layers respectively.

InceptionNet [26], introduced the same year as VGG was special with its Inception Module. This module contained filters of multiple sizes and can extract features at varying scales. ResNet [27], attempts at solving the problem of increasing error rates with an increase in the depth of the architecture. They presented residual blocks with skip connections that let them grow the network



(a) InceptionNet Module (b) Residual Block

Figure 2.2: Building modules from InceptionNet (left) and ResNet (right).

in depth but without the increase in error rate. U-Net [28], is a fully convolutional architecture used for image segmentation, the emphasis in this work was on data augmentation to use data with less number of samples but more efficiency. With Faster-RCNN [29], a novel region proposal network was introduced that shared convolutions with the object detector and resulted in faster object detection. YOLO [30], streamlined the object detection architectures further. Instead of using a sliding window like prior works, they proposed detection by regression which required evaluating the image only once. More network architectures such as DenseNet [31], CenterNet [32] etc. and their variations were introduced over time.

An interesting group of architectures that emerged over the years are Generative Adversarial Networks (GANs) [33]. They follow a two player zero sum game, where a generator produces samples and a discriminator distinguishes the sample between generated or training data. Some of the applications in vision include image generation, video generation, image to image translation, text to image generation among others [34].

Transformers [35], an encoder-decoder architecture first emerged as an improvement to recurrent neural networks architectures like LSTM [36] and GRU[37] for Natural Language Processing. Popular techniques like BERT[38], GPT[39] and their variations are derived from the use of transformers. Its strong dominance in vision applications is also demonstrated in [40, 41, 42, 43].

The latest advances in field of computer vision include Segment Anything Model (SAM) [44], a promptable zero-shot image segmentation model; Stable Diffusion [45], an open source latent

diffusion based text to image generator; Dall-E [43] another text to image generator based on transformers;DINO [46], a transformer based object detector.

2.2 Synthetic Data

This section briefly touches upon the past works that utilized synthetic data, going from approaches to create a synthetic dataset, to some known virtual datasets and finally towards works where synthetic datasets improved the underlying models.

Firstly, the creation of synthetic data is important and there are a variety of ways to do it. Some of the noticeable works in the field use engines like RAGE engine as in [7, 47, 48, 49]. Other engines like Unreal Engine and Unity Engine are also useful, as used in [50] and [51] for generating corresponding virtual datasets. In [48] evaluation of visual perception tasks was performed on a custom benchmark created from commercial software by using detouring[52].

For the purpose of detecting and tracking pedestrians, [11] offered a sizable open-source synthetic dataset for substituting real datasets. In [12] Virtual KITTI dataset was published that showed frameworks that have been pretrained on synthetic data and fine-tuned on actual data surpass those that have only been trained on only synthetic or real data. Synthia [51] is a collection of synthetic photos in an urban setting of a virtually generated city used to boost networks for semantic segmentation.

The performance of deep learning models in many computer vision tasks has been improved by synthetic data. [47] improved their detector by utilizing only synthetic images to train their model. [7] proposed the SSIM Embedding cycle GAN for their synthetic dataset and improved model permanence on crowd counting. [14] demonstrated that their Gaussian Process-based framework, trained on synthetic data, outperformed other domain adaptation techniques relying on real data. [8] exhibited that the performance of convolutional architectures for classification significantly improved with synthetic chest X-ray scans for a medical image classification task. [50] enhanced the performance of a person re-identification model by presenting a synthetic dataset with 100 virtual human subjects under various lighting circumstances.

2.3 Multiple Object Tracking

Some of the recent Tracking paradigms and techniques are discussed in this section. These include a wide variety of techniques including, graph models, Siamese networks, convolutional networks like CenterNet, YOLO etc. and even transformers.

A tracker that employs social and grouping behaviours inside of a graph model and formulates the tracking as a least cost flow optimization issue was proposed in [53]. MDNet is a multi-domain learning convolutional neural network architecture proposed in [54] to learn domain-independent features during pretraining and domain-specific information during the tracking.

[55] developed a fully convolutional Siamese network that was trained offline to learn a similarity function. The trained model was then evaluated online during training to locate a template image within a search image, using the embeddings learned in the offline phase. In a similar vein [56] proposed a Siamese Region Proposal Network that consisted of a template and a detection branch, both of which were trained offline. The network formulated tracking as a local one-shot detection task.

Tracktor was developed in [57] and uses bounding box regression to track objects without the need for extra training. A point-based tracking system called CenterTrack was introduced in [58] It employs the CenterNet [59] detector which is conditioned on two consecutive frames and predicts a displacement vector for associating locations of the objects through frames.

FairMOT, another CenterNet based strategy for a multi-task learning approach for detection and re-identification, was introduced in [60] In this method, fairness was included to handle the accuracy rivalry between the two tasks. As a consequence, both tasks are treated equally by an impartial network, which has no adverse effect on accuracy.

As a recurrent extension to YOLO[30], in [61] they added an LSTM stage. The training or architecture involves three phases: pertaining of convolutional layers, training of object proposal module and training of recurrent LSTM module.

[62] presented TransTrack an attention-based query-key scheme motivated by transformers [35]. It uses attention to track objects, the framework generates two sets of queries containing information for new coming objects and information for maintaining tracklets. Also following the tracking

by attention paradigm for joint detection and tracking, Trackformer was introduced in [63]. Attention is computed between frame features, tracks and object queries to provide bounding boxes and identities.

2.4 Metrics used for tracking

This section discusses some of the popular metrics associated with the task of tracking objects.

Multiple Object Tracking Accuracy (MOTA) and Multiple Object Tracking Precision (MOTP) were introduced in [64], these metrics provide an intuitive measure of performance at detecting and keeping trajectories.

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t}$$

m_t, fp_t, and mme_t are the number of misses, false positives and mismatches respectively.

Identity-based precision (IDP), recall (IDF) and F1 (IDF1) were introduced in [65]. These metrics were calculated by matching the computed tracks with the ground truths one-to-one with minimal frame mismatch. Standard Precision, Recall and F1 were built upon this matching.

$$IDF1 = \frac{2IDTP}{2IDTP + IDFP + IDFN}$$

Higher Order Tracking Accuracy (HOTA) which is quite intuitive, was introduced in [66]. It is the geometric mean of the percentage of aligning detections (Det.A) and average alignment of trajectories (AssA).

$$HOTA = \sqrt{DetA \cdot AssA}$$

2.5 Conclusion

In the chapter, a brief history of computer vision was discussed. From early models like Perceptron to architectures more commonly used in computer vision were presented, along with some of the latest advances in Computer vision such as Dall-E and DINO. The discussion followed with an overview of the literature on the use of synthetic data and Multiple Object Tracking.

Chapter 3

Synthetic Data as a Supplement for Training Deep Learning Models

This is the manuscript for the paper titled ”Synthetic Data as a Supplement for Training Deep Learning Models” which is under preparation

Abstract

Obtaining training data for machine learning models can be challenging. Capturing or gathering the data, followed by its manual labelling, is an expensive and time-consuming process. In cases where there are no publicly accessible datasets, this can significantly hinder progress.

In this paper, we analyze the similarity between synthetic and real data. While focusing on an object tracking task, we investigate the quantitative improvement influenced by the concentration of the synthetic data and the variation in the distribution of training samples induced by it. Through examination of three well-known benchmarks, we reveal guidelines that lead to performance gain. We quantify the minimum variation required and demonstrate its efficacy on prominent object-tracking neural network architecture.

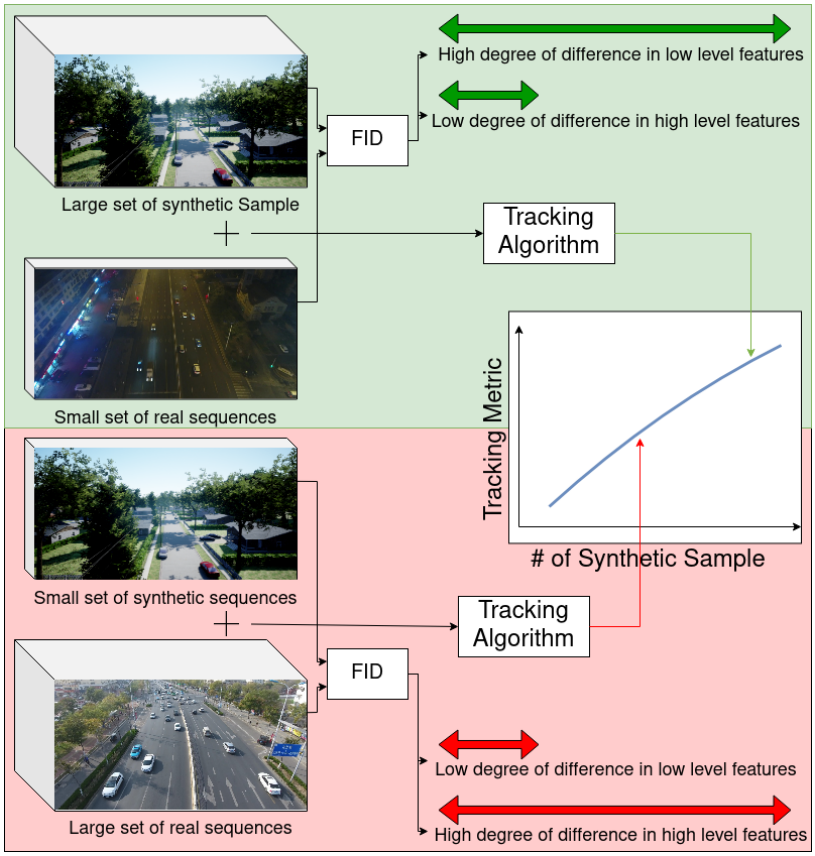


Figure 3.1: A combination of synthetic and real datasets with more synthetic samples and higher variance in distribution (top) outperforming another combination with a lower number of synthetic samples and lower variance in distribution (bottom).

3.1 Introduction

The process of data collecting is not without challenges, requiring substantial investments of time and resources to obtain labelled samples of high quality. Given the resource and capital costs associated with data acquisition, it is more pragmatic and cost-effective to utilize publicly available datasets that have already been published. In practice, however, straightforward application of such datasets may not always be feasible or effective due to a variety of potential challenges or constraints such as biases. One potential solution to these challenges is utilizing synthetic data as a supplement to real-world datasets. By supplementing real-world data with synthetic data, researchers can overcome the limitations inherent to traditional data sources [13], thereby enhancing the overall quality and utility of their datasets.

The utilization of synthetic data has been extensively documented in recent literature, as evidenced by multiple works [7, 8, 14]. However, much of this prior research has focused on domain adaptation techniques. This paper examines the impact of the direct use of synthetic data on the performance of machine learning models. In a preliminary step, we analyze the Frechet Inception Distance (FID) [15] between the synthetic and the real sequences for three benchmark datasets. Building upon the patterns observed, we form clusters for both low and high level features which makes us inquisitive about the impact of these clusters on the performance of the models, if affected then by how much and why? Next, we examine the impact of different concentrations of photo-realistic sequences on training for the three benchmarks and two rendered datasets one of which is generated by us using a game engine. We demonstrate that the use of synthetic images during training can positively affect performance. Also, we discuss instances where the clusters from our preliminary analysis provide an additional stimulus in the form of a gain or drop in performance. We quantify the variation and provide design guidelines for creating synthetic datasets used to train object-tracking models.

The approach we propose is both simple and straightforward, involving the direct utilization of synthetic data without the need for additional domain adaptation steps during training. We justify this approach by viewing the domain adaptation step as a potentially costly and extra procedure when dealing with an already challenging task.

Our proposed strategy aligns with prior studies that have incorporated synthetic data into their training procedures without domain adaptation. However, our approach differs significantly from those studies. For instance, the Virtual KITTI dataset [12] involves a two-step process where pre-training is performed on the virtual data followed by fine-tuning on real data. The MOTSynth challenge [11] encourages training on synthetic data only and testing on real data, without any use of the latter during the training phase. In contrast, our approach involves fine-tuning models on an amalgamation of both actual and synthetic data, thereby improving tracking performance. To the best of our knowledge, this technique has not been previously explored in the literature.

In this paper, we present the following contributions:

- We investigate the efficacy of integrating synthetic data with real data for improving the performance of Multiple Object Tracking.

- We conduct a comprehensive analysis of our experimental results and offer insights on the synergistic effects of using synthetic and real data. Drawing from our observations, we formulate a set of general recommendations for the generation and incorporation of synthetic data to enhance model performance.

In the following sections, we conduct a literature review of previous works involving synthetic data and Multiple Object Tracking, followed by an analysis of the similarity between real and synthetic data, our experimentation on different datasets and finally a discussion of the results and conclusions drawn from it.

3.2 Related Works

This section delves into the various methodologies employed by researchers to generate synthetic data for computer vision tasks. The discussion highlights the value of synthetic data in improving deep learning models' performance through techniques such as domain adaptation and pre-training. Additionally, we examine the different tracking techniques that incorporate various concepts, including behavioural models, graph models, convolutional architectures, and transformers, to achieve robust and accurate tracking performance.

3.2.1 Synthetic Data

To study the impact of synthetic data, the creation of the dataset is important. [47], [48], [49], [7] use RAGE for generating their corresponding virtual datasets. Detouring [52] was employed in [48] to create synthetic a benchmark from commercial software and evaluate visual perception tasks. In [50] a dataset of virtual human subjects under different illumination conditions was developed using Unreal Engine. In [51], Unity Engine was used to develop a dataset for semantic segmentation. Some of the known publically available virtual datasets include Synthia[51] a collection of synthetic images in an urban environment of a virtual city, MOTSynth [11] a large open-source synthetic dataset for pedestrian detection and tracking, Virtual KITTI dataset[12] a synthetic adaptation of popular KITTI Vision benchmark [70].

Various studies demonstrate the efficacy of using synthetic data for enhancing the performance

(a) A sample frame from the UAVDT benchmark [67] (real)

(b) A sample frame from the VisDrone dataset [68] (real)

(c) A sample frame from the dataset generated using AirSim [69] (synthetic)

Figure 3.2: Samples of UAV datasets used in our experiments for vehicle tracking.

of deep learning models in various computer vision tasks. In [47] utilized only synthetic images to train their model, which outperformed the model trained on actual images in object classification. Wang et al.[7] simulated a crowd in their GCC dataset and proposed the SSIM Embedding cycle GAN for counting crowds in the wild. Sindagi et al.[14] demonstrated that their Gaussian Process-based framework, which was trained on synthetic data, outperformed other domain adaptation techniques that relied only on real data. H. Zunair and A. Hamza [8], utilized domain adaptation to generate synthetic chest X-ray scans and showed that when used as supplementary data during training, the performance of convolutional architectures for classification improved. In [50] a synthetic dataset was used along domain adaptation to improve the performance of a person re-identification task.

3.2.2 Multiple Object Tracking

L. Taixé et al. introduced a tracker that uses social and grouping behaviours inside a graph model formulating the tracking as a minimum cost flow optimization problem[53]. H. Nam and B. Han proposed MDNet[54] a multi-domain learning convolutional neural network framework that learns domain-independent features during pretraining and domain-specific information during the tracking.

L. Bertinetto[55] trained a fully convolutional Siamese network to learn a similarity function in an offline manner to be evaluated online during training to locate a template image within the search image using the strong embeddings learned in the offline phase. B. Li[56] proposed a Siamese Region Proposal Network consisting of a template and a detection branch which are trained offline and correlational maps for feature extraction, the tracking is formulated as a local one-shot detection task.

P. Bergman and T. Meinhardt introduced tracktor[57] that exploits bounding box regression of an existing object detector, without any additional training required for tracking objects. Zhou presented a point-based tracking framework called CenterTrack[58] that uses CenterNet[59] detector conditioned on two consecutive frames that also predicts a displacement vector for associating positions of the objects through frames.

Y.Zhang et al. introduced FairMOT[60], also a CenterNet[59] based technique for a multi-task

learning approach for detection and re-identification. In this technique, the competition for accuracy between the two tasks was addressed by introducing fairness. This results in an unbiassed network which treats both tasks equally thus, it doesn't affect their accuracy adversely.

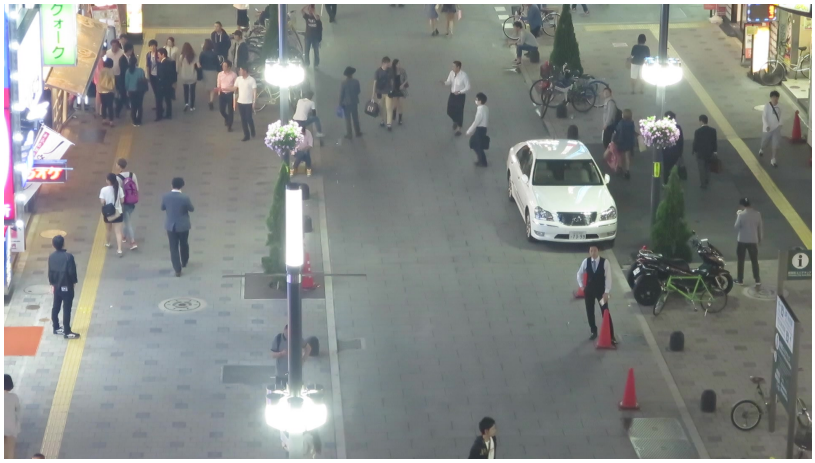
G. Ning et. al proposed ROLO[61] a recurrent extension of YOLO[30] architecture by adding an LSTM stage, training involves three phases pertaining of convolutional layers, training of object proposal module and training of recurrent LSTM module.

P. Sun et al. introduced TransTrack[62] an attention-based query-key scheme inspired by transformers[35] that uses attention to track objects, their framework generates two sets of queries containing information for new coming objects and information for maintaining tracklets. T. Meinhardt et al. introduced Trackformer[63] following the tracking by attention paradigm for joint detection and tracking, attention is computed between frame features, tracks and object queries to output bounding boxes and identities.

3.3 Synthetic Dataset & Observations

In this section we touch upon the synthetic dataset that we created and a publically available synthetic dataset for supplementing the real dataset. We also discuss our key observations when comparing these synthetic datasets to real video sequences using Fréchet Inception Distance (FID) [15].

Our experimental setup utilizes synthetic video sequences generated with the AirSim plugin [69] for Unreal engine. Details on the generation of the dataset are discussed in Section 3.4.1. Along with this new dataset, we use two published Unmanned Aerial Vehicle (UAV) benchmarks for the detection and tracking of vehicles, UAVDT benchmark [67] and VisDrone dataset [68] as real sequences. Samples for the synthetic dataset, the UAVDT benchmark and the VisDrone dataset are provided in Figures 3.2c, 3.2a & 3.2b respectively. To eliminate bias due to the domain and task, and ensure the generalization of the insights, we additionally use another pair of real and synthetic tracking datasets with people tracking in place of vehicle tracking as the objective. For this purpose, we use MOT17 [71] and MOTSynth [11] as real and synthetic datasets respectively. Samples of both datasets can be found in Figures 3.3a and 3.3b respectively.



(a) MOT17 (real)

(b) MOTSynth (synthetic)

Figure 3.3: Samples of MOT datasets used in our experiments. (a) Real image from MOT17. (b) Synthetic image from MOTSynth.

Fréchet Inception Distance (FID) is a quality measure first introduced in [15], for capturing the similarity of the images generated by GANs, this metric also correlates with human judgement. FID score for 2 identical images is 0, and for 2 identical sets of images or videos is close to 0. It increases as the visual similarities between the two images or sets of images reduce as depicted in Figure 3.4. A synthetic sequence with similar lighting, camera angle and elevation as the real sequence results in a relatively lower FID in contrast to another real sequence that has different lighting, camera angle and elevation. We use it to estimate the degree of similarity between synthetic and real images for each pair of real and synthetic sequences.

We computed the Fréchet Inception Distance for three combinations of real and synthetic datasets

Figure 3.4: FID computation example for low level features between AirSim generated video sequence (top) with a similar looking (bottom-left) and a different looking (bottom-right) real video sequence from UAVDT dataset. The computed value towards the green end depicts similarity between the two sequences, and contrary to that towards the red side depicts visual dissimilarity.

from the first pooling layer features (FID64), the second max pooling features (FID192), the pre-auxiliary classifier features (FID768) and the final average pooling features (FID2048)[72]. The computations from the second max pooling (FID192) and the final average pooling (FID2048) features do not add more information or echoes that the first max pooling features (FID64) and the pre-auxiliary classifier features (FID768) already express. Thus, we only use FID scores obtained from the first max pool layer features and the pre-auxiliary classifier features for the low level features and the high level features respectively. The FIDs for all synthetic and real datasets are plotted as heatmaps in Figure 3.5.

In the heatmaps, each row is a real sequence and each column is a synthetic sequence from corresponding real and synthetic dataset pairs. Within the heat maps for low level features 3.5a, the combination of UAVDT benchmark and AirSim generated dataset shows patterns of higher FID scores for some real sequences while most of the real sequences have a relatively lower FID Score. The other 2 dataset combinations only observe a relatively low FID score for all real and synthetic sequence pairings. Contrasting to this, the heatmaps for high level features 3.5b, show more patterns

of high FID scores for real sequences in all three dataset combinations. Interestingly, a pattern for high FID score is also noticeable for synthetic sequences in the MOT17 and MotSynth datasets pairing. We discuss these patterns further in this section.

These initial observations from the heatmaps motivate us to define rigid clusters based on the FID computations. We cluster sequences under 3 categories namely, lower, moderate or higher degrees of difference in low or high level features. This clustering is required to isolate features on the basis of similarity and measure their impact on the performance of the training process. We later use these clusters in further sections for experimentation and discussion.

For the clustering, we create a range between 0 (the minimum achievable distance) and the maximum calculated FID across datasets with an additional buffer. We fit all the sequences to this range and normalize it to get a range between 0 and 1. This normalized scale is divided into three parts using 0.3 and 0.6 as the division points. The sequences that fall under the first, the second and the third segment are termed as sequences with lower, moderate or higher degrees of difference respectively.

3.3.1 FID for low level features

The FIDs obtained after first pooling layer features for the sequences from the UAVDT benchmark and the AirSim generated dataset, sequences with a higher degree of difference have an average of 32.80 ±2.60, the same for sequences having a moderate degree of difference is 18.93 ±2.33 and finally the sequences with a lower degree of difference have an average of 7.80 ±2.93. We observe that all sequences in the VisDrone dataset compared to the synthetic dataset generated using AirSim have a lower degree of difference for low level features with an average of 7.40 ±2.31. A similar observation is made for the sequences from the MOT17 and the MOTSynth datasets, all the sequences lead to an average of 8.66 ±3.48 thus falling under a lower degree of difference for low level features. The FIDs for all synthetic and real sequences are visually represented in Figure 3.5a, where the green regions represent the lower degree of difference, yellow-orange shades depict the moderate degree of difference and dark orange-red represents the higher degree of difference.

3.3.2 FID for high level features

On the basis of the clustering scheme discussed earlier in this section and the FIDs obtained from the pre-auxiliary classifier features for UAVDT and AirSim generated sequences, we cluster sequences among low, medium or high degrees of difference for high level features among real data. The average values for clusters are 1.60 ±0.05, 2.13 ±0.19 and 2.65 ±0.12 respectively. With the same synthetic dataset when FID values are calculated along the VisDrone dataset the clusters obtained have 1.72 ±0.01 for lower degree of difference, 2.07 ±0.22 for a moderate degree of difference and 2.76 ±0.21 for a higher degree of difference in high level features. When calculating the FIDs with higher level features for MOT17 and MOTSynth datasets, we obtain 1.75, 1.99 ±0.16 and 2.72 for lower, moderate and higher degrees of differences. Unlike other real and synthetic dataset combinations, we also observe a pattern for synthetic sequences for MOT17 and MOTSynth. The average values for lower, moderate and higher degrees of difference in high level features are 1.57 ±0.09, 2.01 ±0.20 and 2.68 ±0.18 respectively. We visualize these FID computations in the form of a heatmap in Figure 3.5b.

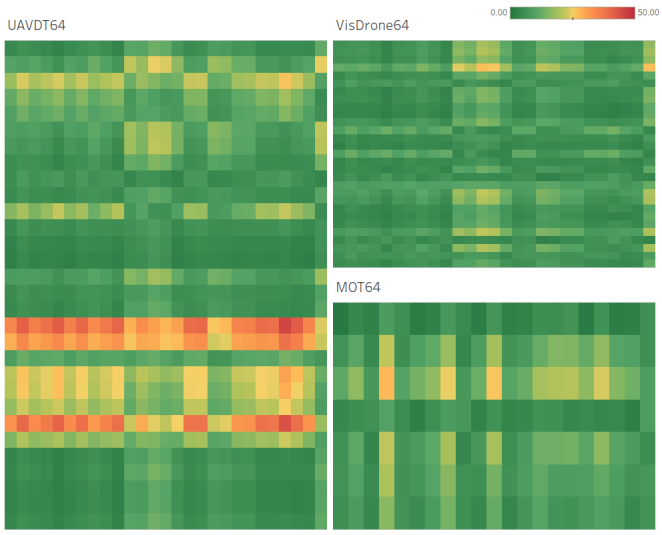
3.3.3 Objectives

With the derived insights and our objective of impact investigation of synthetic data as a supplement in combination with real data, we aim to answer the following questions:

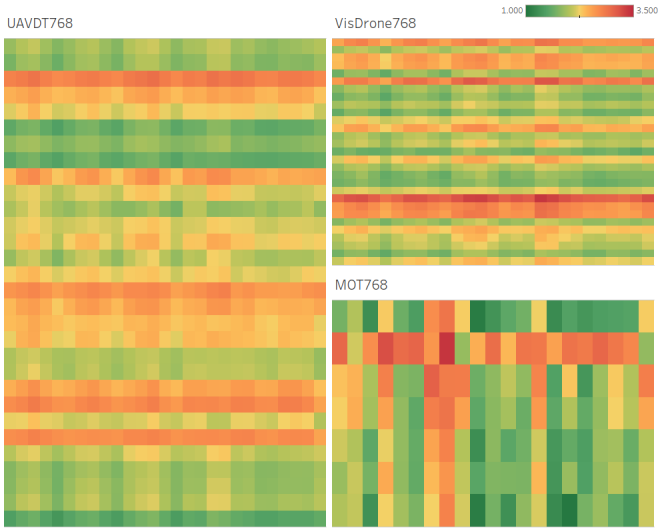
- How effective is the use of synthetic data when supplementing a real dataset?
- What is the impact of different real-synthetic concentrations on the performance metric?
- What are the characteristics of the synthetic data that drive this change, the concentration, the degree of diversity in information brought by the synthetic samples or both factors?

3.4 Experiments

In this section, we discuss in detail the synthetic and the real datasets that we use, our strategy to answer the questions that were raised in the previous section and our trials.



(a) FID64: Fréchet Inception Distance obtained from first pooling features. Left: UAVDT and AirSim, Right-Top: Visdrone and AirSim, Right-Bottom: MOT and MOTSynth. Each row represents a real sequence and each column represents a synthetic sequence.



(b) FID768: Fréchet Inception Distance obtained from pre-auxiliary classifier features. Left: UAVDT and AirSim, Right-Top: Visdrone and AirSim, Right-Bottom: MOT and MOTSynth. Each row represents a real sequence and each column represents a synthetic sequence.

Figure 3.5: FIDs heatmaps for low level features (a) and high level features (b). Dark green represents lower degree of difference, dark red represents higher degree of difference, and the shades in between represent moderate degree of difference in lower/higher level features.

3.4.1 Datasets

As already discussed briefly in Section 3.3, we generate a set of synthetic video sequences using the AirSim plugin in Unreal engine. To generate the simulated video sequences, we load the environment with a simulated drone and dictate its flight trajectory by a set of three-dimensional points in the simulation environment transmitted through APIs. We assimilate input from the camera mounted on the virtual drone and detect vehicles within the field of view of the drone using another pair of APIs to save frames and annotation to a local storage device. We also alter weather conditions across different flight paths to create a diverse set of simulated video sequences. In total, we generated 25 sequences exhibiting different weather conditions, providing a diverse range of scenarios for evaluation.

For the real dataset we use the Unmanned Aerial Vehicle Benchmark [67] (UAVDT) which is a collection of video sequences captured by drones. This benchmark dataset offers sequences with various conditions for illumination, camera viewpoint and elevation. To ensure that our experimentation is not limited to a single dataset, we also conduct tests on another UAV detection and tracking dataset called VisDrone [68]. It contains both city and country environments with annotations for many objects in various weather and lighting conditions. Since our synthetic dataset only had information about vehicles, we rank all the sequences on the most number of vehicles in the scene and only considered the top 30 videos which had the most number of vehicles for our vehicle tracking experiments.

For the extensiveness of our experiments, we use another pair of real and synthetic datasets. MOT17 [71] dataset which is a pedestrian detection and tracking dataset with video sequences having different viewports, camera movements and weather conditions. For the synthetic part, we used the MOTSynth [11] dataset which was created for pedestrian detection, tracking and segmentation and contains frames generated using a rendering game engine. We only required a limited number of sequences according to our experiment setup and a random selection of 21 sequences is used to serve as the training set.

3.4.2 Strategy

We use models trained only on real datasets as baseline models to compare and evaluate against the results obtained from models discussed further in our training strategy. In our strategy, we keep the total number of real and synthetic training sequences constant, that is the number of real sequences available for training. We then substitute real sequences with synthetic sequences. We focus on the substitution and not on the addition of new data for two reasons. First, additional training data will lead to unfair evaluation as the new dataset will have more training samples when compared to the baseline model. Second, substitution creates an artificial scarcity of data enabling us to evaluate the impact of synthetic data when the actual data is insufficient or missing. For these reasons, we formulate an approach to break down the datasets into different-sized folds such that, a bigger chunk from the real dataset has a complementary smaller fold in the synthetic dataset and vice-versa. The combined dataset always accounts for the same number of total video sequences as originally in the training set for the real dataset. For the vehicle tracking experiments, we use ratios 1:5, 1:2, 1:1, 2:1 and 5:1 between real and synthetic data i.e. when there are 5 real sequences we use 25 synthetic sequences, 10 real and 20 synthetic sequences and so on. For people tracking experiments, we use 1:6, 1:2, 2:1 and 6:1 as the ratios.

We use multiple folds for each concentration of real-synthetic combination to understand the consistency of the change in the tracking metric with the real-to-synthetic data ratio. Within the folds, we vary the number of sequences with lower, moderate and higher degrees of difference for low-level and high-level features as discussed in the previous section. In our experiments, each fold is denoted by a lowercase letter.

3.4.3 Training

For our experiments, we train FRCNN[73] network for object detection and ResNet50 [74] model for re-identification, together these two models are used in combination as described in the Tracktor[57] technique. The datasets, both real and synthetic are aimed for detection and tracking and not for re-identification. To allow training of re-identification models on these sequences we create a re-identification dataset from the given frames. We crop the frames where bounding boxes

are present and use these crops for tracked objects as a re-identification dataset. We use the described setup of an object detector and a re-identifier with different folds of the training set as discussed in Section 3.4.2. Models trained on different folds are evaluated using HOTA[66] and IDF1 [65] as the calculative measures for assessing performance. The metrics are calculated using trackeval [75]. The results of public detection from different manifestations of the Tracktor on the UAVDT benchmark and the AirSim generated dataset are reported in Table 3.1, on VisDrone dataset and AirSim generated dataset are reported in Table 3.2 and on MOT17 and MOTSynth are reported in Table 3.3. Further, we discuss these results in Section 3.5.

Also, we extend the applicability of synthetic datasets to transformer-based architectures. We select the Transtrack [62] architecture, which is an encoder-decoder framework with a ResNet-50 [74] backbone network. We train the models on MOT17 and MOTSynth datasets, using the same concentrations and folds as used for the Tracktor experiments. Results are reported in Table 3.4 and further discussed in Section 3.5.

3.5 Discussion

Tables 3.1 and 3.3 show a significant increase in performance measure when synthetic data is included in the training set against the benchmark that only contains all real data. The improvement is up to a 14% increase for the UAVDT benchmark and up to a 10% increase for the MOT17 dataset. Also, Table 3.2 shows an increase up to 4% was achieved for the VisDrone dataset. Table 3.4 shows an increase up to 7% for the MOT17 dataset on a transformer-based architecture. There is a positive correlation between the percentage of synthetic data in the training set and the performance measure for the UAVDT benchmark and the VisDrone dataset. The performance increase for the MOT17 dataset is moreover constant and is not affected by changes in dataset concentrations on Tracktor but we again observe the correlation between the number of synthetic samples and the tracking metric on transformer-based architecture. We further discuss each dataset individually under the following subsections.

Real Set		Synthetic Set							
Size	**Fold**	**Size**	**Fold**	**IDP↑**	**IDR↑**	**DetA↑**	**AssA↑**	**IDF1↑**	**HOTA↑**
5	a	25	a	84.232	83.500	**77.950**	63.984	83.864	70.489
5	b	25	a	84.877	83.538	75.616	63.979	84.202	69.405
5	c	25	a	78.229	78.535	67.812	59.057	78.382	63.134
5	d	25	a	**90.251**	**87.227**	77.166	**70.266**	**88.713**	**73.504**
5	e	25	a	85.501	84.279	72.327	62.985	84.885	67.313
5	f	25	a	85.632	83.054	76.823	64.676	84.323	70.378
10	g	20	b	88.740	82.318	73.588	67.232	85.409	70.252
10	h	20	b	82.459	81.612	68.126	62.729	82.033	65.235
10	i	20	b	87.303	81.945	69.367	64.190	84.539	66.584
15	j	15	c	90.029	79.524	66.662	66.109	84.451	66.259
15	k	15	c	87.484	80.520	67.165	63.453	83.858	65.163
20	l	10	d	81.393	78.686	64.573	61.158	80.016	62.722
20	m	10	e	89.997	76.785	62.950	65.291	82.868	64.004
20	n	10	f	85.122	75.669	62.724	63.897	80.118	63.146
25	o	5	g	81.287	75.073	60.347	60.683	78.057	60.378
25	p	5	h	82.752	77.801	62.214	61.078	80.200	61.513
25	q	5	i	81.746	78.741	64.828	61.166	80.216	62.832
25	r	5	g	86.904	69.093	54.706	62.014	76.982	58.104
25	s	5	h	83.274	74.501	61.816	62.026	78.644	61.744
25	t	5	i	82.871	78.444	63.305	61.988	80.597	62.432
30	u	0	NA	82.085	75.486	61.183	60.948	78.647	60.921

Table 3.1: Results for Tracktor technique trained on datasets with different concentrations of UAVDT benchmark (real) and AirSim generated dataset (synthetic). Column Size denotes the number of sequences and Fold denotes which fold was used.

3.5.1 UAVDT

We observe a direct link between the performance measure and the percentage of the synthetic dataset in the overall training set, by increasing the number of synthetic samples we notice an increase in the HOTA metric. It is highest when we use twenty-five synthetic samples and five real ones, and lowest when use twenty-five real and five synthetic samples across a number of folds. Also, among folds comprised of five real and twenty-five synthetic sequences, the HOTA metric is highest when the training set includes sequences with a higher degree of difference for low level features. Additionally, in the folds consisting of twenty-five real and five synthetic sequences, we notice that the HOTA metric reduces when the folds are constituted from sequences with low or moderate degrees of difference for low level features. All experiments are reported in Table 3.1.

Real Set		Synthetic Set							
Size	**Fold**	**Size**	**Fold**	**IDP↑**	**IDR↑**	**DetA↑**	**AssA↑**	**IDF1↑**	**HOTA↑**
5	a	25	a	69.957	<u>72.991</u>	<u>64.611</u>	**57.841**	71.442	**60.701**
5	b	25	a	63.117	65.636	63.323	50.193	64.352	55.975
5	c	25	a	68.029	70.809	59.921	54.752	69.391	56.779
5	d	25	a	68.278	70.847	**65.364**	55.616	69.539	<u>59.961</u>
5	e	25	a	65.193	67.949	64.281	52.798	66.543	57.834
5	f	25	a	<u>71.052</u>	**74.107**	61.694	<u>57.558</u>	**72.547**	59.241
10	g	20	b	63.448	66.078	63.482	50.621	64.736	56.271
10	h	20	b	69.845	72.378	62.859	56.169	71.089	59.043
10	i	20	b	67.547	70.272	63.569	54.147	68.882	58.307
15	j	15	c	66.078	68.449	61.602	51.998	67.242	56.239
15	k	15	c	67.924	70.180	64.041	54.747	69.034	58.902
20	l	10	d	65.904	67.146	61.679	51.545	66.519	56.053
20	m	10	e	65.603	67.782	64.136	51.997	66.675	57.462
20	n	10	f	67.121	68.683	63.215	53.425	67.893	57.779
25	o	5	g	65.085	65.854	62.107	50.922	65.467	55.953
25	p	5	h	65.430	66.037	61.531	50.750	65.732	55.557
25	q	5	i	65.541	66.822	62.856	51.431	66.176	56.511
25	r	5	g	69.491	67.100	62.294	52.595	68.275	56.943
25	s	5	h	**72.598**	71.483	62.676	55.977	<u>72.036</u>	58.948
25	t	5	i	64.689	65.665	62.643	50.480	65.173	55.885
30	u	0	NA	64.324	64.601	61.910	49.864	64.462	55.252

Table 3.2: Results for Tracktor technique trained on datasets with different concentrations of Visdrone dataset (real) and AirSim generated dataset (synthetic). Column Size denotes the number of sequences and Fold denotes which fold was used.

3.5.2 VisDrone

Our findings indicate an increasing trend, albeit with a few deviations for the VisDrone and AirSim datasets. The performance of models trained on different folds generally increases, except for the folds where five sequences are synthetic the models perform worst than the benchmark but the performance increases gradually as the percentage of synthetic data increases. Another deviation is remarked, where models trained on folds with 20 synthetic sequences perform better than the models trained on folds with synthetic data but the latter still outperforms the benchmark model.

With the FIDs analysis for low level features (Section 3.3.1) as the foundation, it is hard to come to conclusions as all sequences for this dataset combination fall under a low degree of difference. Drawing on the insights derived from the FID analysis for high level features (Section 3.3.2), experiments with folds having 5, 10 or 15 real sequences, the fold having the most number of sequences with a lower degree of difference for high level features outperforms the rest of the folds in that category.

Real Set		Synthetic Set							
Size	**Fold**	**Size**	**Fold**	**IDP↑**	**IDR↑**	**DetA↑**	**AssA↑**	**IDF1↑**	**HOTA↑**
3	a	18	a	48.893	63.291	40.617	52.277	55.168	45.904
3	b	18	b	49.996	59.487	41.928	48.766	54.330	45.069
3	c	18	c	53.866	59.708	**44.196**	49.285	56.636	46.547
3	d	18	d	**55.581**	62.286	43.842	51.946	**58.743**	47.559
3	e	18	e	48.792	61.824	41.173	50.731	54.540	45.552
3	f	18	f	50.057	64.044	40.420	53.327	56.193	46.312
3	g	18	g	52.789	61.614	43.340	50.591	56.861	46.726
7	h	14	h	46.250	61.112	39.961	49.537	52.652	44.250
7	i	14	i	52.175	63.552	42.574	52.785	57.304	47.294
7	j	14	j	48.708	62.582	41.035	52.501	54.780	46.313
14	k	7	k	47.634	62.714	40.470	50.411	54.143	45.080
14	l	7	l	49.877	65.935	41.136	**56.22**7	56.793	**47.927**
14	m	7	m	50.716	63.499	42.412	53.109	56.392	47.333
18	n	3	n	48.730	63.345	40.822	52.392	55.084	46.142
18	o	3	o	50.245	64.282	41.520	53.084	56.403	46.873
18	p	3	p	48.210	63.303	40.821	52.738	54.735	46.289
18	q	3	q	50.598	**66.076**	41.154	56.023	57.311	47.906
18	r	3	r	49.462	65.915	41.347	55.496	56.515	47.806
18	s	3	s	50.404	65.135	41.934	54.193	56.830	47.570
18	t	3	t	50.398	63.233	41.706	53.768	56.090	47.234
21	u	0	NA	34.597	46.205	40.230	32.198	39.567	35.883

Table 3.3: Results for Tracktor technique trained on datasets with different concentrations of MOT17 dataset (real) and MOTSynth dataset (synthetic). Column Size denotes the number of sequences and Fold denotes which fold was used.

Real Set		Synthetic Set							
Size	**Fold**	**Size**	**Fold**	**IDP↑**	**IDR↑**	**DetA↑**	**AssA↑**	**IDF1↑**	**HOTA↑**
3	a	18	a	62.653	53.443	45.824	49.286	57.683	46.968
3	b	18	b	44.166	72.288	39.513	67.894	54.831	51.514
3	c	18	c	66.111	57.044	47.974	51.432	61.244	49.268
3	d	18	d	67.937	55.626	45.120	53.142	61.168	48.695
3	e	18	e	67.314	64.160	49.847	55.861	65.699	52.523
3	f	18	f	**74.162**	59.827	**51.015**	56.889	**66.228**	53.463
3	g	18	g	62.410	57.089	43.517	53.589	59.631	47.928
7	h	14	h	44.227	73.213	40.035	67.432	55.143	51.738
7	i	14	i	62.208	64.166	48.555	54.669	63.172	51.261
7	j	14	j	63.459	66.120	49.082	58.896	64.762	53.421
14	k	7	k	43.836	78.055	40.616	68.205	56.142	52.385
14	l	7	l	45.196	79.390	41.596	68.821	57.600	53.242
14	m	7	m	60.775	67.998	48.565	59.332	64.184	53.390
18	n	3	n	44.370	78.334	41.759	68.752	56.651	53.378
18	o	3	o	42.178	77.860	40.276	66.858	54.716	51.656
18	p	3	p	44.261	77.375	40.184	69.770	56.311	52.751
18	q	3	q	43.894	78.164	41.590	66.904	56.218	52.510
18	r	3	r	43.388	79.442	40.974	69.318	56.124	53.073
18	s	3	s	60.466	59.020	44.717	56.860	59.734	50.124
18	t	3	t	44.689	**81.538**	41.186	**69.973**	57.735	**53.523**
21	u	0	NA	43.207	79.424	41.177	67.884	55.967	52.633

Table 3.4: Results for TransTrack architecture trained on datasets with different concentrations of MOT17 dataset (real) and MOTSynth dataset (synthetic). Column Size denotes the number of sequences and Fold denotes which fold was used.

3.5.3 MOT17

We observe up to 10% increase in HOTA metric for supplementing the MOT17 dataset with the MOTSynth dataset. However, unlike the previous two benchmarks where an increasing trend was observed, our examination reveal about a constant increase in performance measure invariant of the real-synthetic concentrations throughout the experiments.

Guided by the FIDs analysis of high level features, amongst the folds with 3 real sequences, the HOTA metric is the least when trained on samples with higher degree of difference for high level features in comparison to when these samples are excluded. The same is observed in folds with 7 real sequences. This phenomenon becomes hazy for folds with 18 real scenarios. Also to note, the performance metric improves when sequences with higher degree of difference are excluded from the folds comprising synthetic data.

Experiments with the TransTrack architecture show a similar result, an almost constant trend for the HOTA metric. However, the trend is clearly visible in the IDF1 metric. The performance of the model is directly correlated with the amount of synthetic data in the training dataset. The fold including the sequences with a higher degree of difference for high-level features, always performs the worst among the folds of the same size.

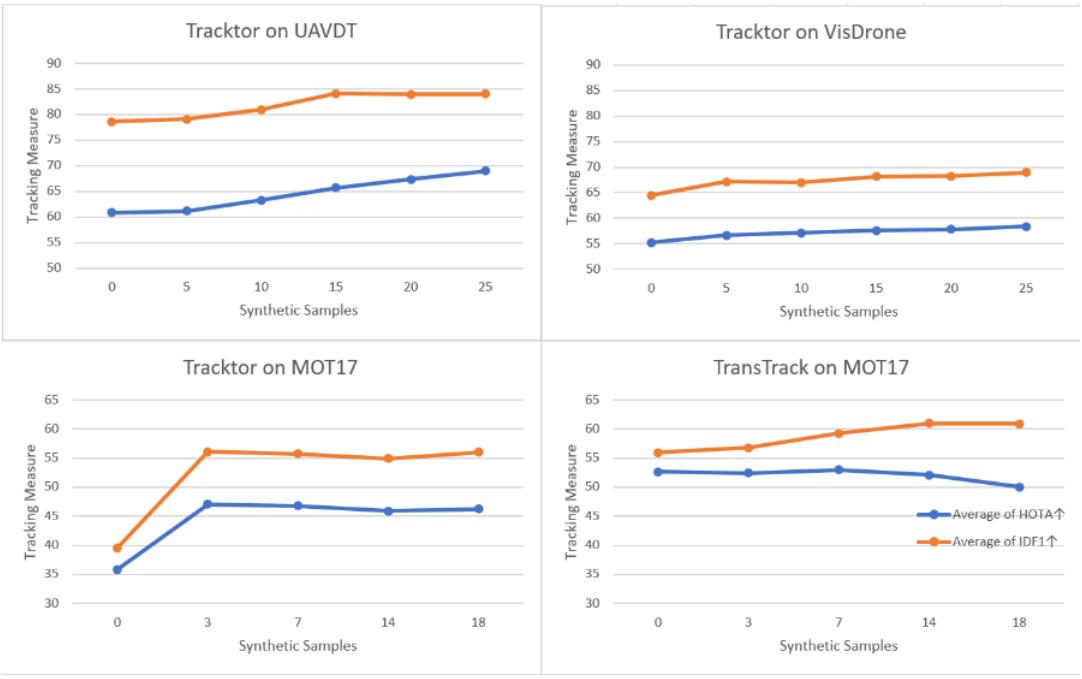


Figure 3.6: Change in tracking performance on increasing the concentration of synthetic samples in the training set for Tracktor on UAVDT benchmark (top-left), VisDrone dataset (top-right) and MOT17 dataset (bottom-left); for TransTrack on MOT17 dataset (bottom-right)

3.5.4 Guidelines

Considering the key insights derived in this section we can deduce that by using synthetic data, one can increase the performance of a model. We derive the following principles from our experiments.

- When synthetic data is used in orders of magnitudes of real data a performance can be anticipated. In our experiments, the increase in performance was up to 15% when synthetic data accounted five times more than the actual video sequences.
- The performance improvement is higher when the variance in low-level features is high. In our experiments, we clustered sequences with values greater than 0.6 on our scale (FIDs greater than 30 units calculated from first pooling layer features) as a high degree of difference for low-level features. The presence of these sequences resulted in a better performance.
- The increase in performance is limited by the variance in high-level features and is recommended to be kept minimal. Our experiments with sequences with values lower than 0.3 on

our scale(under 1.75 units for FIDs calculated from pre-auxiliary classifier features) showed an increased improvement.

3.6 Conclusion

In this study, we investigated the effectiveness of using synthetic data in combination with real data for Single Camera Multi-Object Tracking tasks. We utilized three different datasets and two different tracking techniques to evaluate the impact of using synthetic data. Our results indicate that the inclusion of synthetic data in the training process of deep learning models improves the performance metrics when compared to using real data alone. Furthermore, we also explored the specific aspects of synthetic data that should be emphasized to further enhance the performance of the models.

Our findings suggest that the combination of real and synthetic data can lead to a new paradigm for training deep learning models. While synthetic data has traditionally been used for pre-training or domain adaptation, our study highlights the potential for a simpler technique to complement real data in the training process. We aim to validate the application of synthetic data for solving challenges such as bias mitigation, generalization of outside datasets, and wider applicability of existing datasets in our future works. We believe that this approach can lead to improved performance in a range of computer vision tasks and can pave the way for the development of more sophisticated and accurate models. Overall, this paper contributes to the growing body of research on the use of synthetic data and its potential for enhancing the capabilities of deep learning models.

Chapter 4

Extended Experimentation

Abstract

This chapter presents the experimentation that isn't documented in previous chapters. A qualitative experiment for assessing models trained real data on a synthetic image, the detailed process by which synthetic data was generated for experimentation in Chapter 3 with a preliminary evaluation of models trained in combination with synthetic data. FID heatmaps excluded in from the analysis in 3.3. An attempt to induce FID variation between synthetic and real datasets. Finally, samples of the training datasets.

4.1 Clipart Experiment

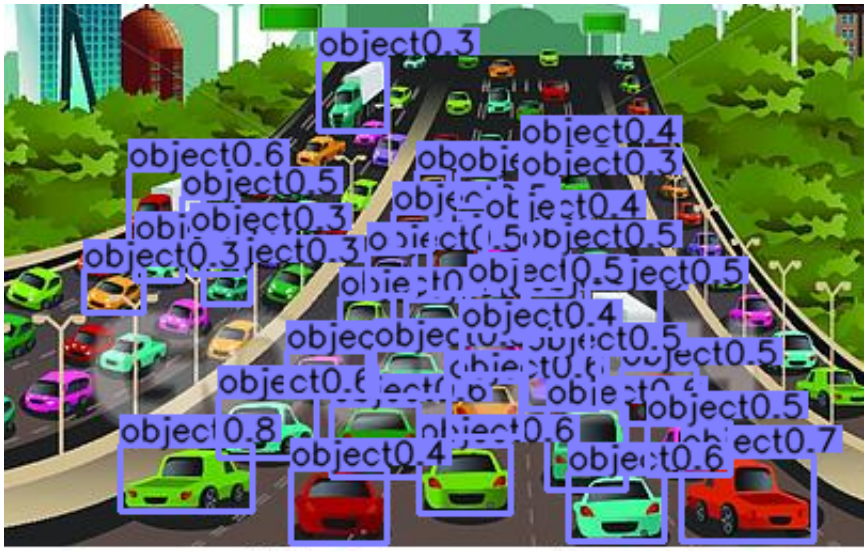


Figure 4.1: Model trained on real datasets successfully detecting in a synthetic scenario. Source: Google Images

A model trained on real data successfully detected vehicles in a synthetic scenario as depicted in 4.1. This raised the question, will a model trained on synthetic data will perform equally well on real data?

4.2 Synthetic data generation with AirSim

AirSim[69] is a cross-platform open-source plugin for Unreal engine to simulate cars and drones, it works as a platform by providing APIs for computer vision and reinforcement learning tasks. The developers tried to provide close-to real-world physics and visuals for autonomous cars and drone training. The inbuilt car and drone APIs can be used to communicate and control the choice of vehicle, the Time of Day API lets the user retrieve or set the time of the day by changing the position of the sun, and the weather API enables or disables different atmospheric conditions such as rain, snow, dust and fog.

The experimental setup utilizes synthetic video sequences generated from a pre-compiled binary AirSimNH environment, which was provided as an additional resource for the AirSim plugin [69] for Unreal engine. This environment simulates a small block in an urban neighbourhood and is

designed to be easily customizable for different use cases. The particular environment was selected as it required minimal adjustments and was readily available for use. The objective is to evaluate the efficacy of synthetic data for the purpose of assessing its impact on object tracking performance, rather than creating a highly realistic simulation.

To generate the simulated video sequences, the environment was loaded with a MultiRotor object as the selected vehicle, and two parallel processes were used. The first process dictated the flight trajectory, while the second process captured the feed from the simulated drone's camera. In order to define the flight path, a connection with the environment using the MultiRotorClient API was established. Once they connected, a set of three-dimensional coordinates were provided to act as the path for the drone to follow. As the first process guides the movement of the simulated drone within the environment, the second process assimilates input from the camera mounted on the virtual drone. This was accomplished by establishing a parallel connection using the VehicleClient API. An inbuilt detection API was also set up to detect vehicles within the field of view of the drone. These processes execute in order to save frames and annotations to local storage.

To create a diverse set of simulated video sequences, the aforementioned steps for different flight paths, and weather conditions were repeated. In total, 25 sequences were generated with an image resolution of 960x540 pixels. The simulated sequences exhibited varying weather conditions, including bright and sunlit conditions, rainfall, and snowfall, providing a diverse range of scenarios for evaluation.

4.2.1 Prelimnary results of training with synthetic data

Experiment	HOTA	Det. Acc.	Ass. Acc.
R30 Det & ReId	0.61	0.61	0.61
R15aS15 Det & ReId	0.66	0.67	0.66
R15aS15 Det & R30 ReId	0.67	0.67	0.67
R15bS15 Det & ReId	0.65	0.67	0.63
R15bS15 Det & R30 ReId	0.66	0.67	0.65
R30S15 Det & ReId	0.61	0.62	0.60
R30S15 Det & R30 ReId	0.62	0.62	0.62
R30 Det & R30S15 ReId	0.61	0.61	0.60
R30 Det & R15aS15 ReId	0.61	0.61	0.61
R30 Det & R15bS15 ReId	0.61	0.61	0.61

Table 4.1: Tracking metrics for test sequences, quantity after *R* represents the number of real sequences and that after *S* represents the number of synthetic sequences. *R30 Det & ReId* is the benchmark with detector and re-identifiers trained on 30 real sequences.

4.3 Clusters made using Preliminary FID analysis

FID Dims		UAVDT			VisDrone			MOT17		
		low	mod	high	low	mod	high	low	mod	high
64	mean	7.80	18.93	32.80	7.40	-	-	5.95	-	-
	std dev	2.93	2.33	2.60	2.31	-	-	3.97	-	-
768	mean	1.60	2.13	2.65	1.72	2.07	2.76	1.75	1.99	2.72
	std dev	0.05	0.19	0.12	0.01	0.22	0.21	-	0.16	-

Table 4.2: FID statistics for different clusters formed as per discussion in Section 3.3

4.4 Excluded FIDs

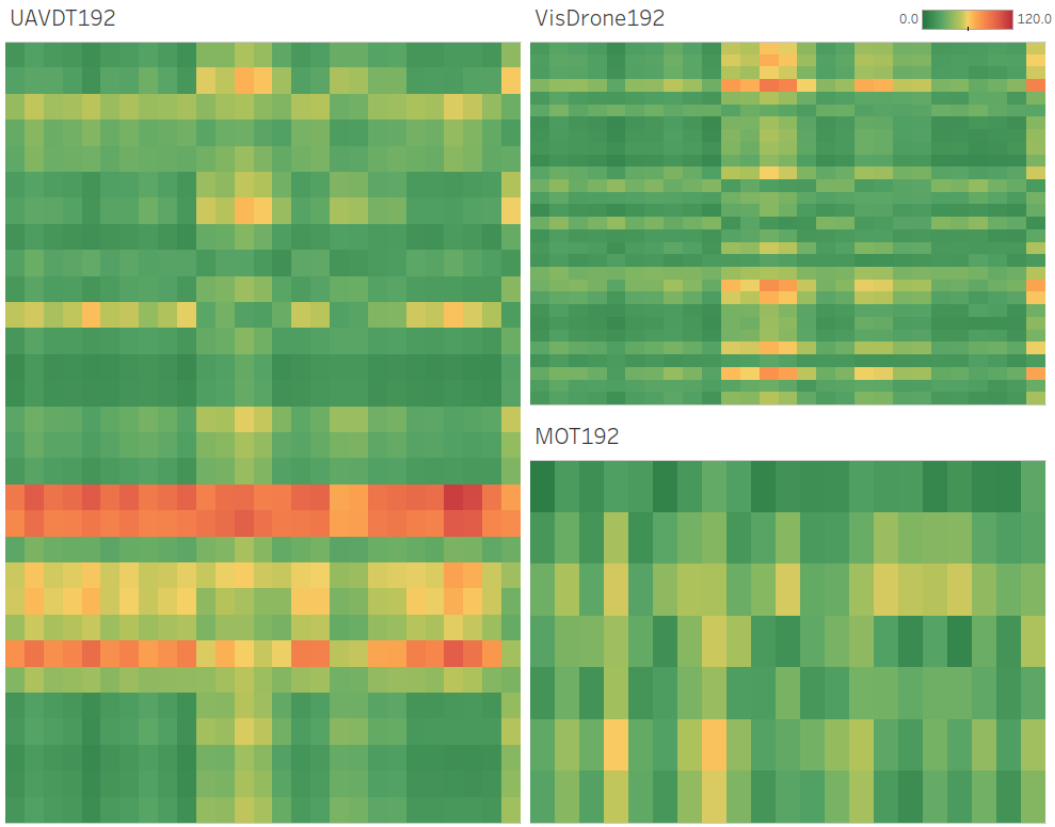


Figure 4.2: FID192: Fréchet Inception Distance obtained from second max pooling layer features. Left: UAVDT and AirSim, Right-Top: Visdrone and AirSim, Right-Bottom: MOT and MOTSynth. Each row represents a real sequence and each column represents a synthetic sequence.

Figure 4.3: FID2048: Fréchet Inception Distance obtained from final average pooling layer features. Left: UAVDT and AirSim, Right-Top: Visdrone and AirSim, Right-Bottom: MOT and MOTSynth. Each row represents a real sequence and each column represents a synthetic sequence.

4.4.1 Failed attempt on inducing FID changes

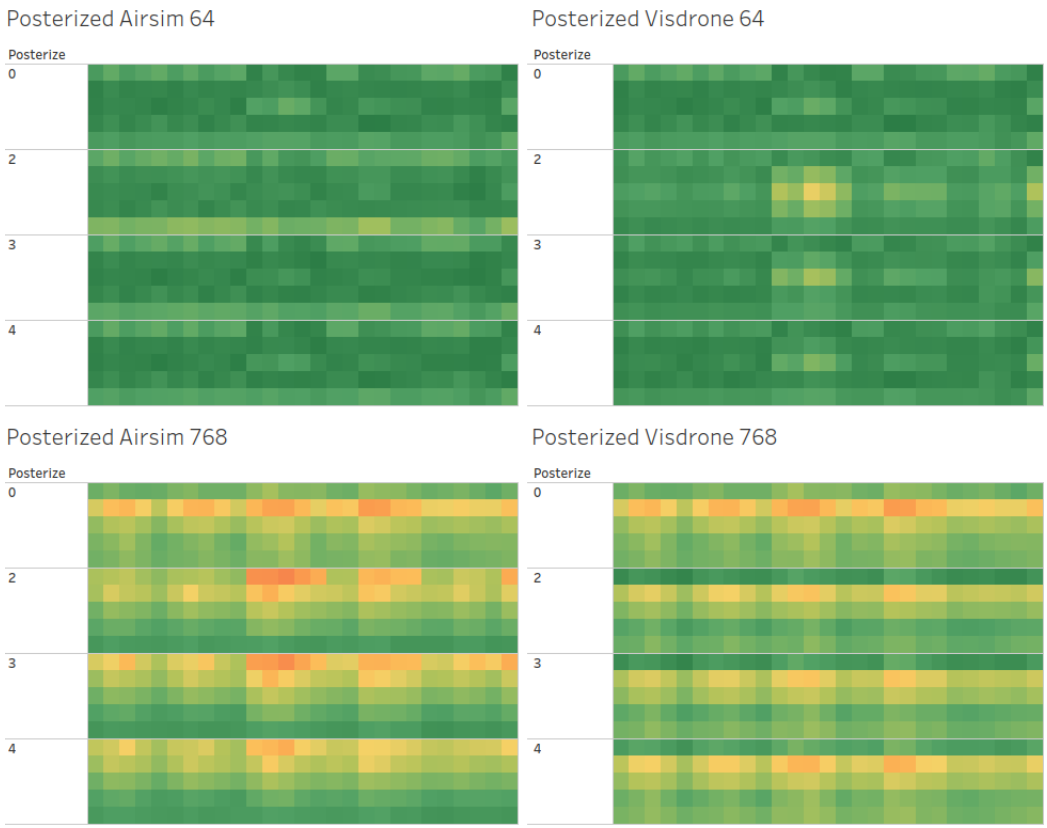


Figure 4.4: FID64 (top) and FID768 (bottom) computations for a fold of 5 sequences from VisDrone dataset with sequences from AirSim generated dataset. Left: Synthetic data is posterized, Right: real data is posterized. Each row represents a real sequence and each column represents a synthetic sequence. Posterizing the sequences didn't bring any significant changes to the FID computation.

4.5 Samples of Sequences

Figure 4.5: Frames from UAVDT dataset

Figure 4.6: Frames from UAVDT dataset (contd.)

Figure 4.7: Frames from UAVDT dataset (contd.)

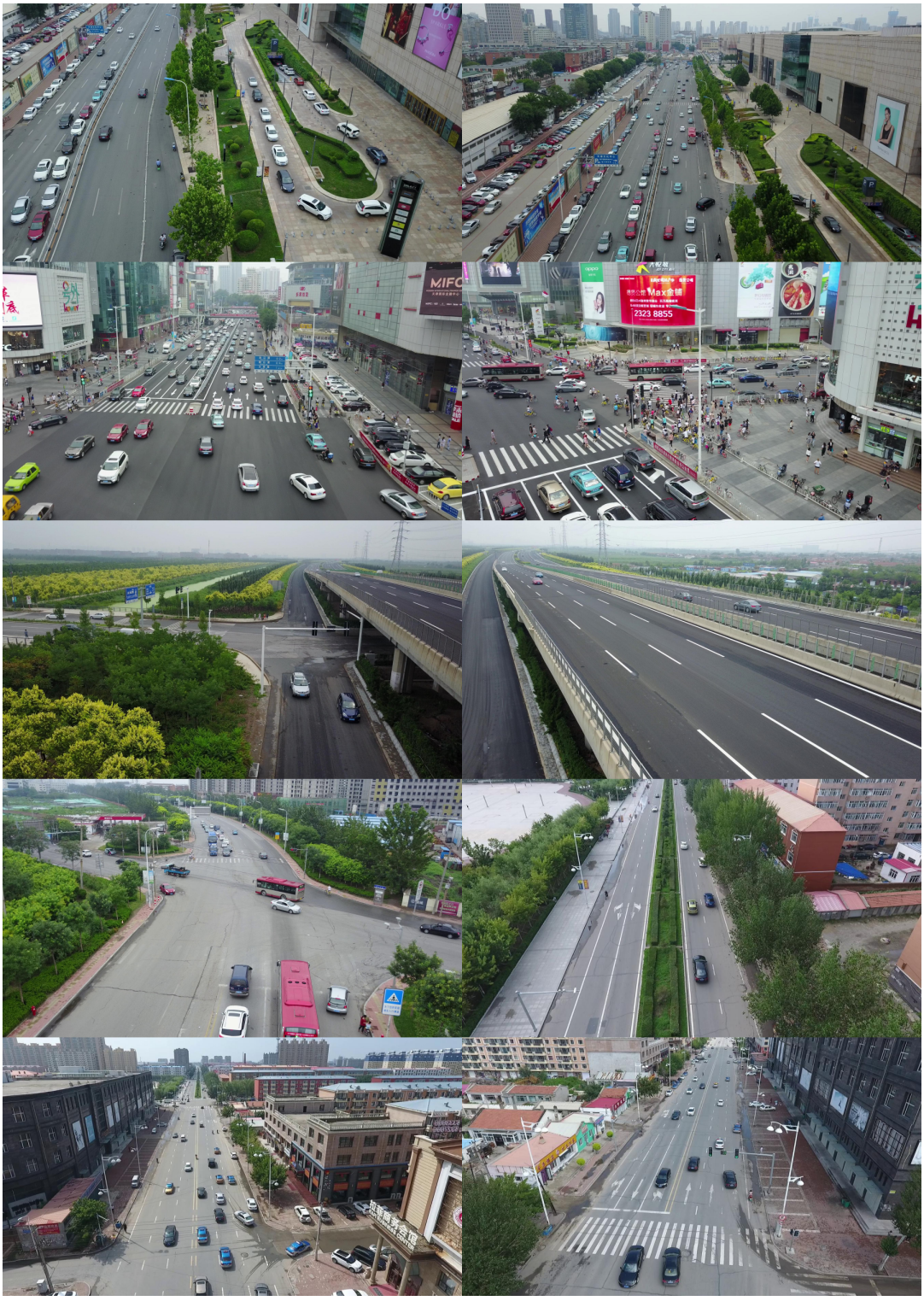


Figure 4.8: Frames from VisDrone dataset sequences used for training

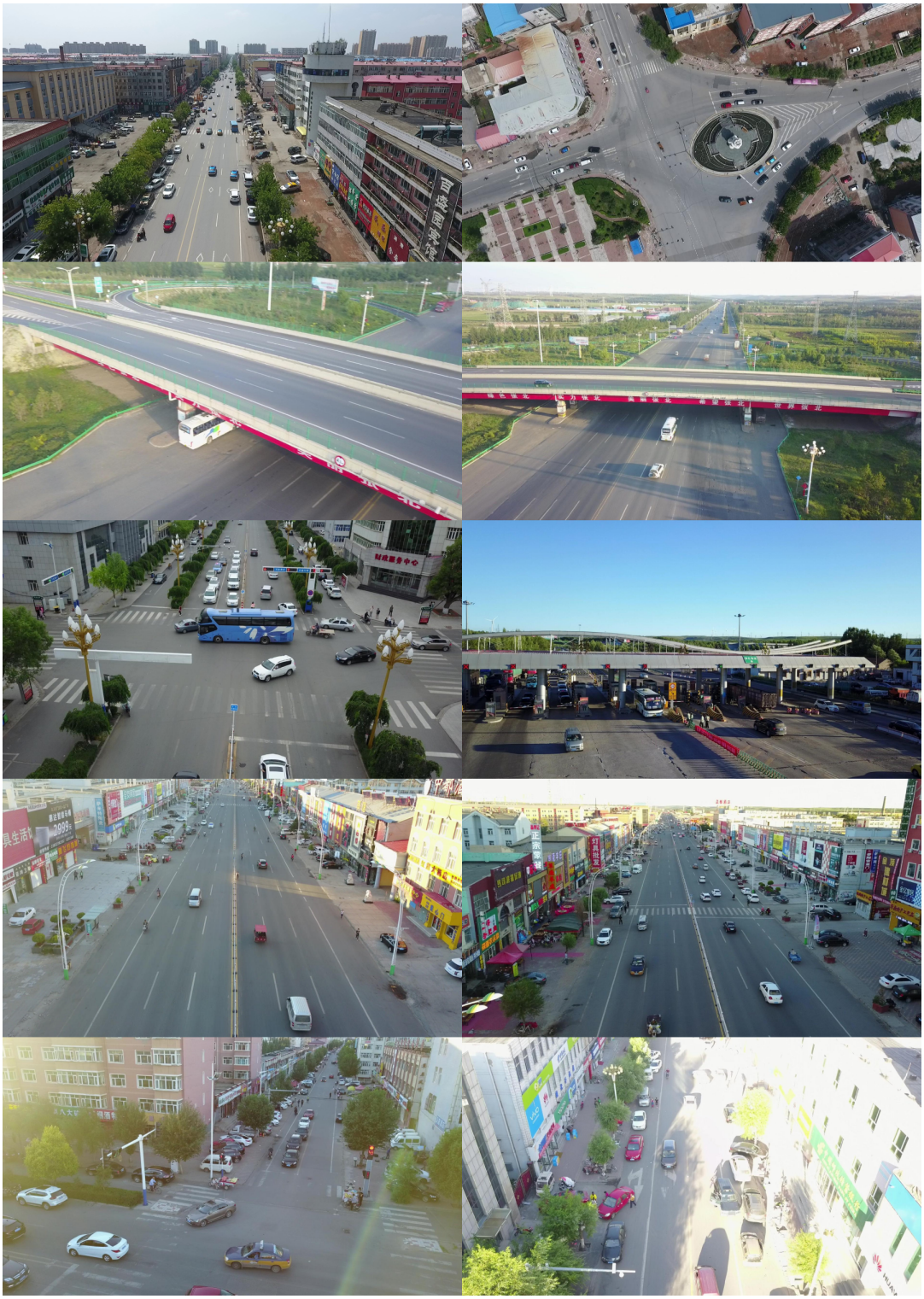


Figure 4.9: Frames from VisDrone dataset sequences used for training (contd.).

Figure 4.10: Frames from VisDrone dataset sequences used for training (contd.)

Figure 4.11: Frames from MOT17 dataset

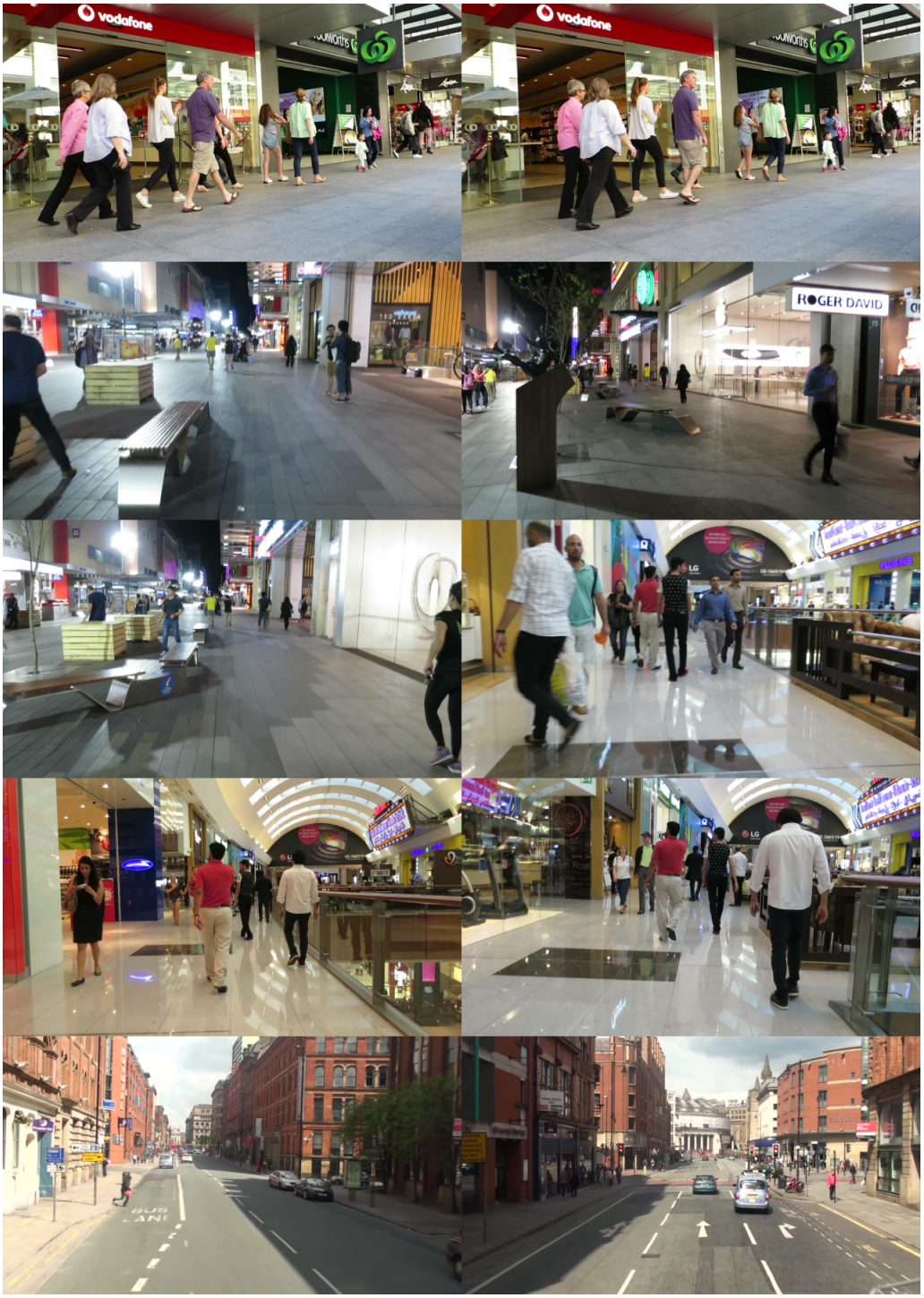


Figure 4.12: Frames from MOT17 dataset (contd.)

Figure 4.13: Frames from MOT17 dataset(contd.

Figure 4.14: Frames from AirSim dataset

Figure 4.15: Frames from AirSim dataset (contd.)

Figure 4.16: Frames from AirSim dataset(contd.)

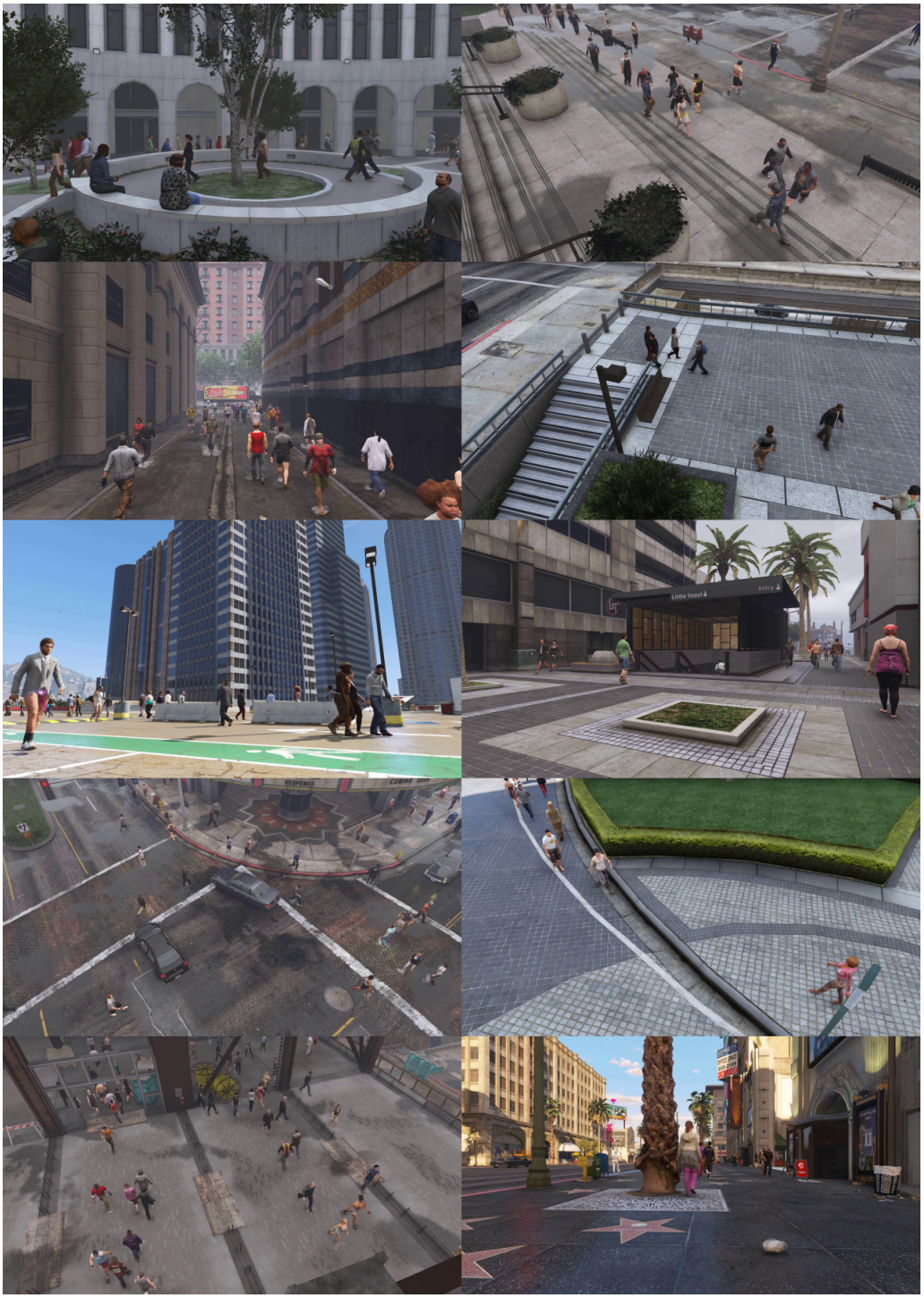


Figure 4.17: Frames from MOTSynth dataset

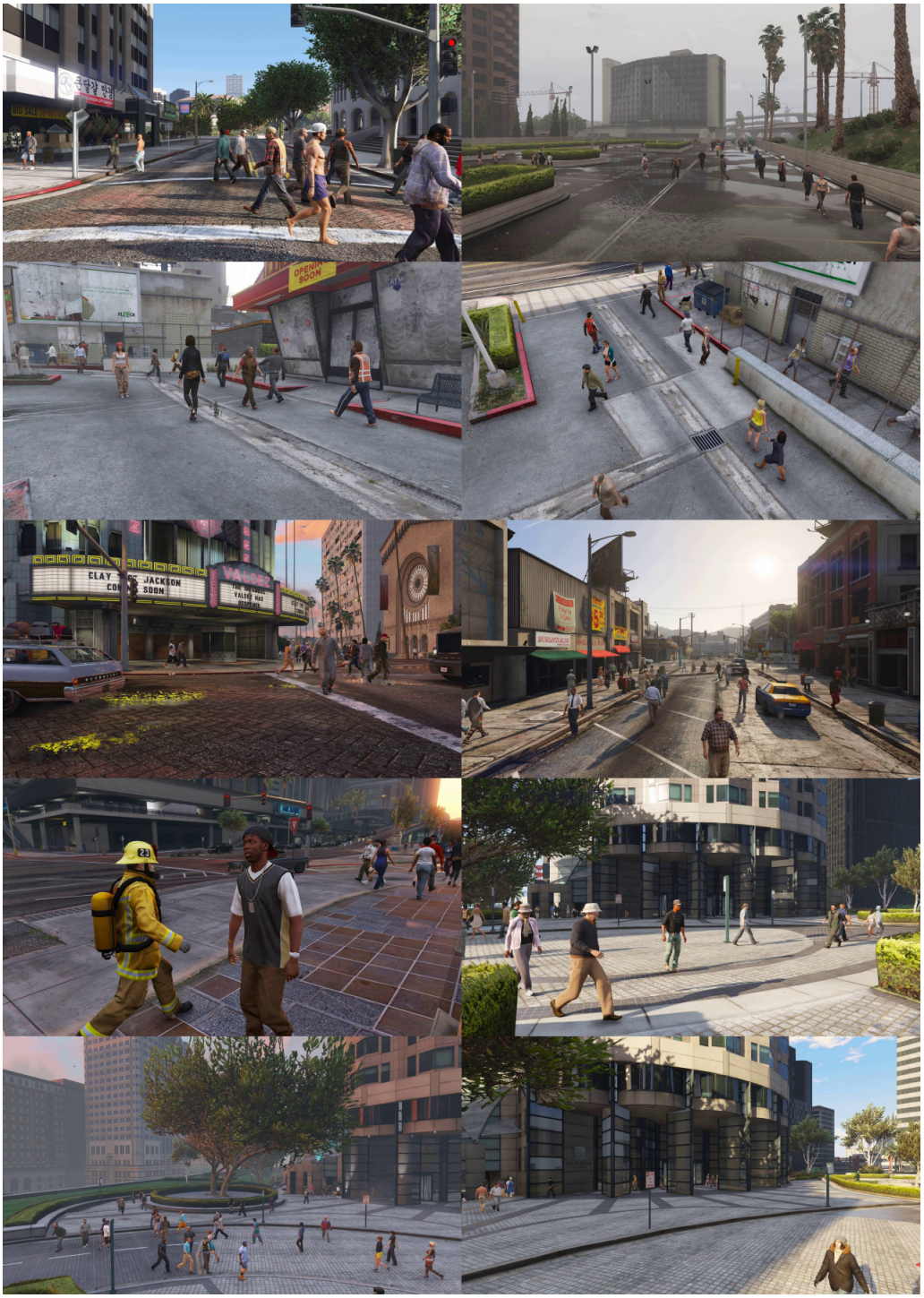


Figure 4.18: Frames from MOTSynth dataset (contd.)

Figure 4.19: Frames from MOTSynth dataset (contd.)

4.6 Conclusion

This chapter presented a qualitative assessment of a model trained on real data for a synthetic image, the process by which synthetic data was generated using AirSim along with a preliminary evaluation of training in combination with synthetic data. FID heatmaps from the second max pooling layer and final average pooling layer features. FID heatmap for sequences with an induced variation. Lastly, samples of the training datasets.

Chapter 5

Conclusion

The primary objective of this study was to assess the efficacy of incorporating synthetic data with real data, a Single Camera Multi-Object Tracking task was selected to see this impact. To this end, two different tracking techniques were employed on three distinct datasets for the goals of vehicle and pedestrian tracking. The impact of including synthetic data in the training process was investigated with two different synthetic datasets, out of which one was generated for this specific purpose. Experimental results suggest that the inclusion of synthetic data as a supplement in the training yields superior deep learning models compared to using solely real data. Moreover, we also examined the specific characteristics of synthetic data that could be emphasized to further enhance the performance of the models.

The findings from our study suggest that the utilization of a combination of real and synthetic data can represent a novel approach to training deep learning models. While synthetic data has typically been utilized for pre-training or domain adaptation, our work highlights the potential of a simpler technique that can supplement real data in the training process. We intend to validate this application of synthetic data in future works, particularly in resolving issues such as bias mitigation, generalization of outside datasets, and the wider applicability of existing datasets. New architectures like SAM [44] opened gateways for new tracking techniques like Track Anything [76], and further works will also be aligned towards assessing the efficacy of synthetic data with foundation and promptable models. More specifically, the scope of applying synthetic data to such techniques.

Our work presents a promising opportunity to improve the performance of computer vision

tasks, paving the way for the development of more advanced and accurate models. Ultimately, this paper contributes to the burgeoning research on the use of synthetic data and its potential for enhancing the capabilities of deep learning models.

Bibliography

[1] H. Garcia-Molina, M. Joglekar, A. Marcus, A. Parameswaran, and V. Verroios, "Challenges in data crowdsourcing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 4, pp. 901–911, 2016.

[2] M. Nashaat, A. Ghosh, J. Miller, S. Quader, C. Marston, and J.-F. Puget, "Hybridization of active learning and data programming for labeling large industrial datasets," in *2018 IEEE International Conference on Big Data (Big Data)*, pp. 46–55, IEEE, 2018.

[3] Wikipedia contributors, "2023 chinese balloon incident — Wikipedia, the free encyclopedia," 2023. [Online; accessed 17-May-2023].

[4] Y. K. Adimoolam, B. Chatterjee, C. Poullis, and M. Averkiou, "Efficient deduplication and leakage detection in large scale image datasets with a focus on the crowdai mapping challenge dataset," *arXiv preprint arXiv:2304.02296*, 2023.

[5] R. Webster, J. Rabin, L. Simon, and F. Jurie, "On the de-duplication of laion-2b," *arXiv preprint arXiv:2303.12733*, 2023.

[6] A. Barbu, D. Mayo, J. Alverio, W. Luo, C. Wang, D. Gutfreund, J. Tenenbaum, and B. Katz, "Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models," *Advances in neural information processing systems*, vol. 32, 2019.

[7] Q. Wang, J. Gao, W. Lin, and Y. Yuan, "Learning from synthetic data for crowd counting in the wild," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8198–8207, 2019.

[8] H. Zunair and A. B. Hamza, "Synthesis of covid-19 chest x-rays using unpaired image-to-image translation," *Social network analysis and mining*, vol. 11, pp. 1–12, 2021.

[9] A. Kortylewski, A. Schneider, T. Gerig, B. Egger, A. Morel-Forster, and T. Vetter, "Training deep face recognition systems with synthetic data," *arXiv preprint arXiv:1802.05891*, 2018.

[10] S. Mishra, R. Panda, C. P. Phoo, C.-F. R. Chen, L. Karlinsky, K. Saenko, V. Saligrama, and R. S. Feris, "Task2sim: Towards effective pre-training and transfer from synthetic data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9194–9204, 2022.

[11] M. Fabbri, G. Brasó, G. Maugeri, O. Cetintas, R. Gasparini, A. Ošep, S. Calderara, L. Leal-Taixé, and R. Cucchiara, "Motsynth: How can synthetic data help pedestrian detection and tracking?," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10849–10859, 2021.

[12] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," *CoRR*, vol. abs/1605.06457, 2016.

[13] K. Baek and H. Shim, "Commonality in natural images rescues gans: Pretraining gans with generic and privacy-free synthetic data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7854–7864, 2022.

[14] V. A. Sindagi, R. Yasarla, D. S. Babu, R. V. Babu, and V. M. Patel, "Learning to count in the crowd from limited labeled data," in *European Conference on Computer Vision*, pp. 212–229, Springer, 2020.

[15] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[16] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, 1943.

[17] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[18] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[19] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[20] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *Lecture notes in computer science*, vol. 3951, pp. 404–417, 2006.

[21] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*, pp. 2564–2571, Ieee, 2011.

[22] M. Bansal, M. Kumar, and M. Kumar, "2d object recognition: a comparative analysis of sift, surf and orb feature descriptors," *Multimedia Tools and Applications*, vol. 80, pp. 18839–18857, 2021.

[23] B. S. Kusumo, A. Heryana, O. Mahendra, and H. F. Pardede, "Machine learning-based for automatic detection of corn-plant diseases using image processing," in *2018 International conference on computer, control, informatics and its applications (IC3INA)*, pp. 93–97, IEEE, 2018.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[28] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241, Springer, 2015.

[29] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[30] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[31] Y. Zhu and S. Newsam, "Densenet for dense flow," in *2017 IEEE international conference on image processing (ICIP)*, pp. 790–794, IEEE, 2017.

[32] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6569–6578, 2019.

[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[34] G. Iglesias, E. Talavera, and A. Díaz-Álvarez, "A survey on gans for computer vision: Recent research, analysis and taxonomy," *Computer Science Review*, vol. 48, p. 100553, 2023.

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[36] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[37] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[38] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[39] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[40] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pp. 213–229, Springer, 2020.

[41] Y. Fang, B. Liao, X. Wang, J. Fang, J. Qi, R. Wu, J. Niu, and W. Liu, "You only look at one sequence: Rethinking transformer in vision through object detection," *Advances in Neural Information Processing Systems*, vol. 34, pp. 26183–26197, 2021.

[42] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12077–12090, 2021.

[43] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-shot text-to-image generation," in *International Conference on Machine Learning*, pp. 8821–8831, PMLR, 2021.

[44] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment anything," *arXiv preprint arXiv:2304.02643*, 2023.

[45] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

[46] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J.-J. Zhu, L. M. shuan Ni, and H. yeung Shum, "Dino: Detr with improved denoising anchor boxes for end-to-end object detection," *ArXiv*, vol. abs/2203.03605, 2022.

[47] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, and R. Vasudevan, "Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks?," *CoRR*, vol. abs/1610.01983, 2016.

[48] S. R. Richter, Z. Hayder, and V. Koltun, "Playing for benchmarks," *CoRR*, vol. abs/1709.07322, 2017.

[49] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," *CoRR*, vol. abs/1608.02192, 2016.

[50] S. Bak, P. Carr, and J. Lalonde, "Domain adaptation through synthesis for unsupervised person re-identification," *CoRR*, vol. abs/1804.10094, 2018.

[51] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3234–3243, 2016.

[52] D. Brubacher, "Detours: Binary interception of win32 functions," in *Windows NT 3rd Symposium (Windows NT 3rd Symposium)*, (Seattle, WA), USENIX Association, July 1999.

[53] L. Leal-Taixé, G. Pons-Moll, and B. Rosenhahn, "Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker," in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 120–127, 2011.

[54] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," *CoRR*, vol. abs/1510.07945, 2015.

[55] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional siamese networks for object tracking," *CoRR*, vol. abs/1606.09549, 2016.

[56] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8971–8980, 2018.

[57] P. Bergmann, T. Meinhardt, and L. Leal-Taixé, "Tracking without bells and whistles," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[58] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points," *ECCV*, 2020.

[59] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.

[60] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "Fairmot: On the fairness of detection and re-identification in multiple object tracking," *International Journal of Computer Vision*, vol. 129, pp. 3069–3087, 2021.

[61] G. Ning, Z. Zhang, C. Huang, X. Ren, H. Wang, C. Cai, and Z. He, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *2017 IEEE international symposium on circuits and systems (ISCAS)*, pp. 1–4, IEEE, 2017.

[62] P. Sun, J. Cao, Y. Jiang, R. Zhang, E. Xie, Z. Yuan, C. Wang, and P. Luo, "Transtrack: Multiple-object tracking with transformer," *arXiv preprint arXiv: 2012.15460*, 2020.

[63] T. Meinhardt, A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer, "Trackformer: Multi-object tracking with transformers," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[64] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, pp. 1–10, 2008.

[65] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part II*, pp. 17–35, Springer, 2016.

[66] J. Luiten, A. Osep, P. Dendorfer, P. Torr, A. Geiger, L. Leal-Taixé, and B. Leibe, "Hota: A higher order metric for evaluating multi-object tracking," *International Journal of Computer Vision*, pp. 1–31, 2020.

[67] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li, W. Zhang, Q. Huang, and Q. Tian, "The unmanned aerial vehicle benchmark: Object detection and tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[68] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, and H. Ling, "Detection and tracking meet drones challenge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021.

[69] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, 2017.

[70] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[71] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking," *arXiv:1603.00831 [cs]*, Mar. 2016. arXiv: 1603.00831.

[72] M. Seitzer, "pytorch-fid: FID Score for PyTorch." https://github.com/mseitzer/pytorch-fid, August 2020. Version 0.3.0.

[73] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015.

[74] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[75] J. Luiten and A. Hoffhues, "Trackeval." https://github.com/JonathonLuiten/TrackEval, 2020.

[76] J. Yang, M. Gao, Z. Li, S. Gao, F. Wang, and F. Zheng, "Track anything: Segment anything meets videos," *arXiv preprint arXiv:2304.11968*, 2023.