Benchmarking Visual Synthetic vs Real data in DL models

Alexey Gruzdev

Abstract

Deep learning models require a large amount of training data in order to achieve satisfactory results in real-life settings. The problem with real data is that it is expensive, requires a lot of time and effort to collect and label, and is prone to human error when manually labeled. Synthetic data holds the potential of solving those bottlenecks. As a result of the domain gap between real and synthetic data, it is unclear how effective synthetic data is for training deep learning models. In this work we are trying to answer this. We do this quantitatively by comparing a deep learning model tasked with computer vision object classification, trained on a synthetic dataset with a model trained on a subset of the well-known ImageNet (real) dataset. In addition, we present a highly efficient scheme for generating synthetic data that is tailored for our purposes, yet can also be applied to other computer vision tasks.

We demonstrate results that outperform the baseline model trained on the ImageNet subset by 13.16%, when trained on a mixed dataset composed of ImageNet and the generated synthetic data. Additionally, we demonstrate that we can replace a large synthetic dataset with 66.66% of the baseline ImageNet data and reach a performance that is almost equal to the baseline. Our results show that synthetic data can be complementary to real data in some cases.

Table of contents

Contents

Abbreviations

Name	**Description**
BG	Background
w/ BG	With background
wo/ BG	Without background

List of figures

List of tables

1. Introduction

Recent breakthroughs in the field of computer vision boosted by supervised deep learning algorithms managed to yield very impressive results in several computer vision tasks. However, in order to achieve robust results in real-world applications, massive amounts of labeled data are required. While most deep learning models implementations, such as Faster RCNN [1] and Mask RCNN [2] are open source, there is a relatively small amount of labeled data available free to the public. Thus a countable number of free datasets, are unable to satisfy a broad range of domain-specific tasks. Data collection and annotation is an expensive, time consuming [3] and error prone process [4]. Therefore, high quality labeled data is a major bottleneck in applied computer vision. Moreover, there are types of annotations where it is nearly impossible to extract labels manually, e.g. 3D bounding box, 3D pose, eye gaze vector or other geometric properties, surface normal maps. On the other hand, data generated from computational algorithms enables not only to generate both simple and complex ground truth labels automatically, but also in a perfectly precise and controllable manner, and even more importantly, at any desired scale.

Synthetic data has been gaining traction in the recent years and it is becoming more and more popular as an alternative to the real-world data. It might theoretically provide infinite amounts of visual data with labels generated by algorithms with minimal human labor involved. A combination of factors leads to seeking training data alternatives both from academia and commercial companies. Factors such as a vast demand for visual, high quality, high variance labeled data, recent photorealism advances in rendering techniques, and of course challenges in real-world data acquisition in scale. Also programmatic control over the data in 3D graphic simulators [5] [6] [7] opens new possibilities.

The idea of synthetic data as a replacement for real-world data leads to questions regarding the effectiveness of it as a training source. In this work, we will investigate and attempt to answer whether it is possible to achieve at least the same performance results when training deep learning models on synthetic data, as with the real-world data.

2. Literature Review

Multiple works were done in the recent years with promising results that demonstrated feasibility of this direction. But it is also evident that the use of synthetic data as a serious competitor to the real-world is just at the very beginning. This fact makes it even a more exciting research objective. As a clear evidence of the growing popularity and recognition of this idea, there is a large number of synthetically generated datasets available online Flying Chairs [8], Virtual KITTI [9] [10], UnrealStereo [11], SYNTHIA [12], GTA V [13], Sim4CV [14], SIDOD [15], Falling Things [16] AirSim [17], CARLA [18] just to name a few. But the above datasets are still in the realm of academic research, rather than real-world applications.

Previous works done in this direction share the same common outline. The methodology is based on the utilization of 3D modeling software and photorealistic rendering capabilities as a source of synthetic visual data generation. The output of this process are images along with the ground truth labels of interest, that are exported automatically. And eventually when it comes to model training and testing, the process is the same as with real-world data.

According to the best of our knowledge, until now the use of synthetic data in the real-world machine learning based mainstream applications, has been more complimentary to real training data, rather than a mainstream single source. In [19] they suggest the use of synthetic data as a means of rare events and classes expansion. In [20] they propose a method for surveillance systems learning to cover huge amounts of pedestrians walking scenarios to cover edge cases, using synthetic data. [10] observe synthetic datasets as a cost-effective alternative and supplement to a real-world, mainly used to evaluate prototypes to improve performance along with real-world datasets. Also [21] mention a common approach to extend real data with synthetic data, which leads to significant improvements.

Naturally there is some degree of skepticism whether synthetic data does the job in the context of machine learning training. This is a relatively novel research path, and both hardware and software technologies required to advance it further, are in constant state of development and improvement. Although in many cases, photo realistically rendered images are indistinguishable of real ones by the human perception system (e.g. many visual effects in movies make use of these technologies), it is not necessarily the case with neural networks. One of the reasons, as [21] states, is that modern rendering engines perform tricks to fool a human eye where obviously neural networks are not taken into account. In [10] they go even further, claiming that it has been shown that synthetic data cannot completely replace the real-world data. As a result, the synthetic data effectiveness remains an open question. The reason for the potential ineffectiveness is the gap between synthetic and real-world data. When a network is being trained on synthetic data and tested on real-world data, a transition (a gap) between 2 completely different distributions, or domains takes place. In order to achieve robust performance, we need this gap to be as small as possible. Several past works [21] [22] mention two well-known approaches to bridge the gap – domain randomization and domain adaptation.

Domain randomization, as introduced in [23], is a strategy to narrow the gap between synthetic and real-world data realms. Based on the hypothesis that a neural network trained on synthetic data within, leads to better generalization of the trained models, and hence improved performance. In [16] [24] [22] they follow this approach and the results seem promising.

Domain adaptation [25] [26] [27] is another way to tackle it by transferring images from synthetic to real-world visual domains. This might be achieved by learning domain-invariant features or learning mapping between the two domains. We hope that photorealism already mimics the real-world close enough for the gap to be minimal.

The well-known ImageNet dataset [28] will serve us as a real dataset reference. The ResNet 50 [29] as the main neural network architecture during the experiments. In [29] they report achieving top-1 accuracy of 77.15% and top-5 accuracy of 94.75% when trained on the full (1,000 classes) ImageNet on ResNet 50.

3. Contribution

Our first contribution is the proposed method of generating a photo realistic, varied synthetic dataset at large scale, designed for object classification computer vision task.

The second contribution is the first of its kind synthetic vs real data benchmarking done to our best knowledge. According to the research done prior and during this work, we have not found any similar

works, where the benchmarking is performed with a well-known in the community real dataset. More than that, we have not found any work focused on synthetic data benchmarking for the object classification computer vision task. Most of the similar works were focused on object detection, semantic segmentation and pose estimation tasks. Object classification is the most fundamental task and unlike segmentation and detection that besides the neural network involve additional algorithms, involve only a neural network.

Third contribution is a set of experiments that quantitatively measure the synthetic data trained model performance over the real data. Even more interestingly, mixed – synthetic and real data train modes with different proportions and combinations.

4. Data Description

4.1. Real Data

4.1.1. General

ImageNet dataset [28] has been selected as a real data reference. The complete ImageNet contains 1,281,167 training and 50,000 validation images. The test set is not publicly available. There are total of 1,000 distinct object classes. All the images are roughly uniformly distributed. So that it makes 1,281 images per class on average for the train set, and 50 images per class on average for the validation set.

Figure 1 ImageNet dataset

4.1.2.Selected Classes and Quantities

From the complete ImageNet, exactly 17 object classes were selected to satisfy the project objectives. The selected classes are: analog clock, digital clock, wall clock, backpack, basketball, soccer ball, cup, computer mouse, computer keyboard, laptop computer, monitor, remote control, dining table, desk, toaster, vase, banana. In all of the selected classes there are total of 1,200 images for train and 50 for validation. So in total there are 17*1,200=20,400 images for train and 17*50=850 images for validation.

Each object class in ImageNet holds a unique ID. For convenience reasons, we refer to them by class names in this work. Table 1 represents the mapping.

ImageNet ID	Class name
n02708093	alarm_clock
n02769748	backpack
n02802426	basketball
n03085013	pc_keyboard
n03179701	computer_desk
n03196217	digital_clock
n03201208	food_table
n03642806	laptop
n03782006	pc_monitor
n03793489	pc_mouse
n04074963	tv_remote
n04254680	soccer
n04442312	toaster
n04522168	vase
n04548280	wall_clock
n07753592	banana
n07930864	cup

Table 1 ImageNet ID to semantic names mapping

	Class name	Image example
1	alarm_clock	

2	digital_clock	
3	wall_clock	
4	backpack	
5	basketball	

6	soccer	
7	cup	
8	pc_mouse	
9	pc_keyboard	

10	laptop	
11	pc_monitor	
12	tv_remote	

13	food_table	
14	computer_desk	
15	toaster	

16	vase	
17	banana	

Table 2 ImageNet samples of selected classes

4.2. Backgrounds

4.2.1. General

As a background for the objects, we decided to use Places dataset [29] with a wide variety of scenes from different contexts. The dataset contains 2.5 million images from 205 scene categories. Each object class was manually mapped to multiple scene categories based on real life possible associations.

Figure 2 Places dataset

4.2.2.Object Class to Scene Category Mapping

The class to scene category mapping from the places dataset was prepared in order to algorithmically merge class image with a background image from the associated set. The mapping was built manually following the common sense. Each scene category is a directory of background images. For each class, a concrete scene category, and a concrete background image from the set are randomly chosen during the merge algorithm.

```
"alarm_clock ": ["atrium/public", "attic", "bedroom", "general_store/indoor", "gift_shop",
            "home_office", "living_room", "bedchamber", ],  # "analog", "clock"
"backpack ": ["atrium/public", "attic", "bedroom", "classroom", "living_room", "office", ],  # "backpack,", "back"
"basketball ": ["arena/performance", "basketball_court/indoor"],  # "basketball"
"pc_keyboard ": ["atrium/public", "bedroom",
            "booth/indoor", "childs_room", "classroom", "computer_room",
            "home_office", "office", ],  # "computer keyboard"
"computer_desk ": ["archive", "art_school", "art_studio",
            "attic", "bedroom", "bow_window/indoor", "childs_room",
            "classroom", "home_office", "office",
            ],  # "desk"
"digital_clock ": ["atrium/public", "attic", "bedroom", "classroom", "gift_shop", "home_office",
            "living_room", "office", "bedchamber", ],  # "digital", "clock"
"food_table": ["archive", "attic", "bakery/shop",
            "balcony/interior", "banquet_hall",
            "bar", "bow_window/indoor", "cafeteria", "coffee_shop",
            "dining_hall", "dining_room", "fastfood_restaurant",
            ],  # "dining", "table,",
"laptop": ["atrium/public", "bedroom", "bedchamber",
            "bookstore", "booth/indoor", "childs_room",
            "classroom", "computer_room", "home_office", "living_room",
            "office", ],  # "laptop,",
"pc_monitor": ["atrium/public", "bedroom", "bedchamber",
            "bookstore", "childs_room", "classroom",
            "computer_room", "home_office", "living_room",
            "office", "television_room", ],  # "monitor"
"pc_mouse": ["bedroom", "bedchamber", "bookstore",
            "booth/indoor", "childs_room", "classroom",
            "computer_room", "home_office", "office", ],  # "mouse,", "computer",
"tv_remote": ["bedroom", "berth", "computer_room", "general_store/indoor",
            "home_theater", "television_room", ],  # remote", "control,",
"soccer": ["athletic_field/outdoor", "baseball_field", "football_field",
```

```
            "soccer_field", "stadium/football", "stadium/soccer", ],  # "soccer",
"toaster": ["cafeteria", "coffee_shop", "dining_room",
            "fastfood_restaurant", "general_store/indoor", "gift_shop",
            "kitchen", "restaurant_kitchen", ],  # "toaster"
"vase": ["artists_loft", "art_school", "art_studio", "cafeteria",
            "dining_room", "general_store/indoor", "gift_shop",
            "kitchen", "restaurant_kitchen", "storage_room", "supermarket", ],  # "vase"
"wall_clock": ["bedchamber", "bedroom", "berth", "general_store/indoor", "living_room",
            "office", ],  # "wall", "clock"
"banana": ["bakery/shop", "banquet_hall", "cafeteria", "candy_store", "coffee_shop",
            "dining_room", "fastfood_restaurant", "kitchen", "restaurant_kitchen",
            "supermarket", ],  # "banana"
"cup": ["bakery/shop", "banquet_hall",
            "bar", "cafeteria", "candy_store", "coffee_shop",
            "dining_room", "fastfood_restaurant", "kitchen", "restaurant_kitchen",
            "supermarket", ]  # "cup"
```

Table 3 class-backgrounds mapping

Class name	**Number of background categories**
alarm_clock	8
backpack	6
basketball	2
pc_keyboard	8
computer_desk	10
digital_clock	9
food_table	12
laptop	11
pc_monitor	11
pc_mouse	9
tv_remote	6
soccer	6
toaster	8
vase	11
wall_clock	6
banana	10
cup	11

Table 4 Number of backgrounds per class

There are on average 8.47 background categories per each object class.

4.3. Synthetic data

4.3.1. General

The synthetic data originates from a professionally created 3D models combined with algorithmically generated images.

Dataset hierarchy consists of {object classes}/{class instances}/{camera viewpoints}. Each image includes exactly one object in the view point frame. Each object class contains a set of class instances – random variations of an object class. Each class instance includes multiple camera viewpoints, where each view point represents a single shot of the object from a predefined angle. Note that, for comparison, in the ImageNet dataset there's only a single randomly sampled viewpoint of any class instance.

4.3.2. Selected Classes and Quantities

Since the goal was to have an intersection of the same object classes between synthetic and real data sets, the object classes selection is exactly the same as the real data presented in Table 5.

	Class name	**Image example**
1	alarm_clock	
2	digital_clock	

3	wall_clock	
4	backpack	
5	basketball	
6	soccer	
7	cup	

8	pc_mouse	
9	pc_keyboard	
10	laptop	
11	pc_monitor	
12	tv_remote	

13	food_table	
14	computer_desk	
15	toaster	
16	vase	
17	banana	

Table 5 Synthetic data samples of selected classes

Each class instance is generated from 26 camera viewpoints. Data separation split is 80%-20% train-test.

Class name	Number of class instances	Number of total images (instances X view angles)
alarm_clock	425	11050
backpack	1075	27950
banana	174	4524
basketball	28	728
computer_desk	678	17628
cup	3743	97318
digital_clock	308	8008
food_table	424	11024
laptop	444	11544
pc_keyboard	104	2704
pc_monitor	877	22802
pc_mouse	497	12922
soccer	853	22178
toaster	81	2106
tv_remote	499	12974
vase	2445	63570
wall_clock	339	8814
TOTAL	**12994**	**337844**

Table 6 Synthetic training data size

Class name	Number of total images
alarm_clock	2763
backpack	6988
banana	1131
basketball	182
computer_desk	4407
cup	24330
digital_clock	2002
food_table	2756
laptop	2886
pc_keyboard	676
pc_monitor	5701
pc_mouse	3231
soccer	5545

toaster	527
tv_remote	3244
vase	15893
wall_clock	2204
TOTAL	**84466**

Table 7 Synthetic validation data size

While performing the initial experiments and error analysis some of the originally generated data has been filtered out and we created a reduced version. It was done both to balance the instances per class distribution and to reduce training execution times. Too similar instances have been removed from the dataset. In addition 2,010 view angles images have been removed from the training dataset. Those samples poorly represented the class instances and potentially have been "confusing" for the model.

All the experiment results in this work are based on the reduced dataset version.

Class name	**Number of class instances**	**Number of total images (instances X view angles)**
alarm_clock	340	8840
backpack	860	22360
banana	140	3640
basketball	23	598
computer_desk	543	14118
cup	874	22724
digital_clock	246	6396
food_table	340	8840
laptop	356	9256
pc_keyboard	84	2184
pc_monitor	702	18252
pc_mouse	398	10348
soccer	683	17758
toaster	65	1690
tv_remote	400	10400
vase	1307	33982
wall_clock	272	7072
TOTAL	**7633**	**196448**

Table 8 Synthetic training data size reduced

4.3.3.Data Characteristics

Viewpoint distribution

The viewpoints include both deterministically and stochastically sampled rotations.

Deterministic rotations. Deterministically sampled rotations from 16 distinct degrees over x and y axis separately, in range [0°, 360°] with interval steps of 25°, with excluded range of [90°, 25°] (see explanation below). This yields a total of 16 images per object.
Stochastic rotations. Randomly sampled 10 degrees over x and y axis together, in range [0°, 360°] with excluded range of [90°, 25°] (see explanation below).
The total number of 26 viewpoints – 26 renders per each class instance.

Excluded degrees explanation: the range of [90°, 25°] represents the bottom of each 3D object in the source dataset. Since statistically such viewpoints are rare and do not present in train/validation dataset, moreover such samples might "confuse" rather then add a valuable information to the model, those were discarded.

Figure 3 Banana from 26 viewpoints

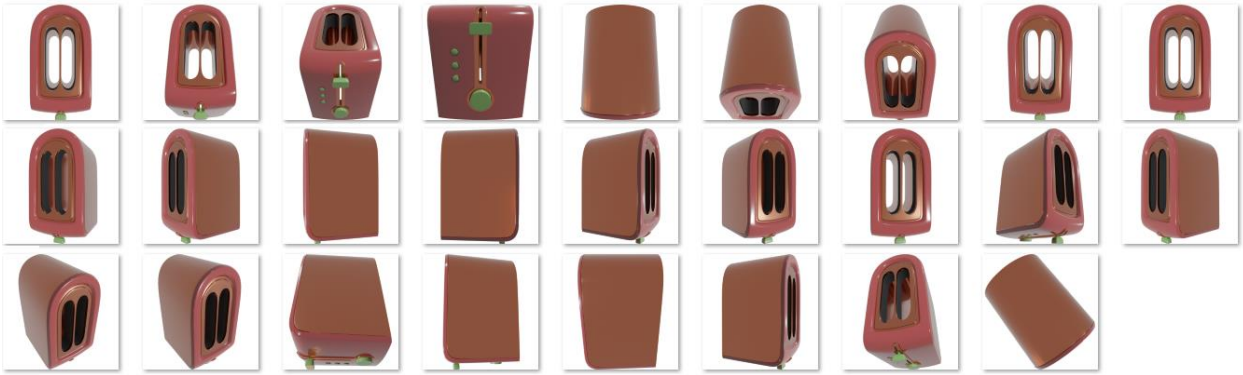

Figure 4 Toaster from 26 viewpoints

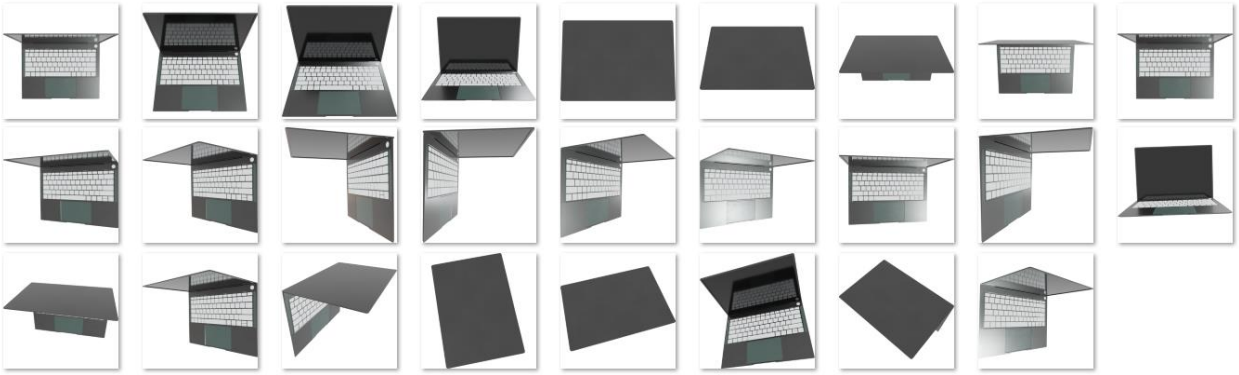

Figure 5 Laptop from 26 viewpoints

Other properties

The camera is auto focused and targeted at the object, while the whole object is covered in full camera frame. Each image is rendered with a transparent background. The lighting used in the scene is a white with ambient properties. This was chosen with a goal of achieving a neutral scene lighting. Each image is rendered at 512x512 resolution.

4.3.4.Object Image with a Background Paste & Copy

The each final input image fed to the model, is a composition of an object image rendered with a transparent background, and an associated randomly sampled background. The goal is as closer as possible to the real world, thus to minimize the domain gap. In addition an augmentation is performed over the object image.

Figure 6 Basketball object images (same instance, different viewpoints) augmented and copy pasted to basketball hall background images

Figure 7 Food table object images (same instance, different viewpoints) augmented and copy pasted to dining hall background images

4.3.5.Synthetic: with a Background vs Without any Background

For the experiment purposes, both object images with and without any background have been tested. Since the merge is performed during the training, the merging phase is controlled in the

code with a parameter. Virtually both of those configurations might be viewed as a 2 separate datasets.

Figure 8 Object image with a background

Figure 9 Object image without any background

4.4. Real/Synthetic Comparison

4.4.1. Quantities

The total size of the synthetic training data is much larger than the real data. However, if counting only the number of different instances per class, it is lower for most of the classes in the synthetic dataset compared to the ImageNet. On average there are 764 instances per class in synthetic dataset (see Table 6), while ImageNet there are 1,200.

4.4.2. Intra Class Variance

Intra class variance is limited in the synthetic data case. Across most of the classes, the class instances within the same class share common visual aspects. This is especially evident comparing it to the ImageNet. In ImageNet the class instances are from the wild, taken from random places around the globe, with a very high level of variance. Figures Figure 10 - Figure 15 demonstrate the intra class variance for both real and synthetic data cases.

4.4.3. Other Variance Aspects

ImageNet data samples also highly vary in camera viewpoints, object locations and orientations relative to the camera. In order to approximate those properties, the synthetic data generator produced multiple viewpoints for each object. In addition, during the augmentation process the algorithm sets random location of an object inside the frame, an object size – simulating the distance from the camera and a random rotation. Also, the lighting conditions in the ImageNet

dataset vary. Synthetic dataset object images intentionally were generated with a standard, neutral lighting. In this work there was no focus on the lighting conditions. Since photos were taken by multiple sources in ImageNet there is high diversity in camera properties like intrinsic parameters, blurry or distorted lenses. Such variations were ignored as well.

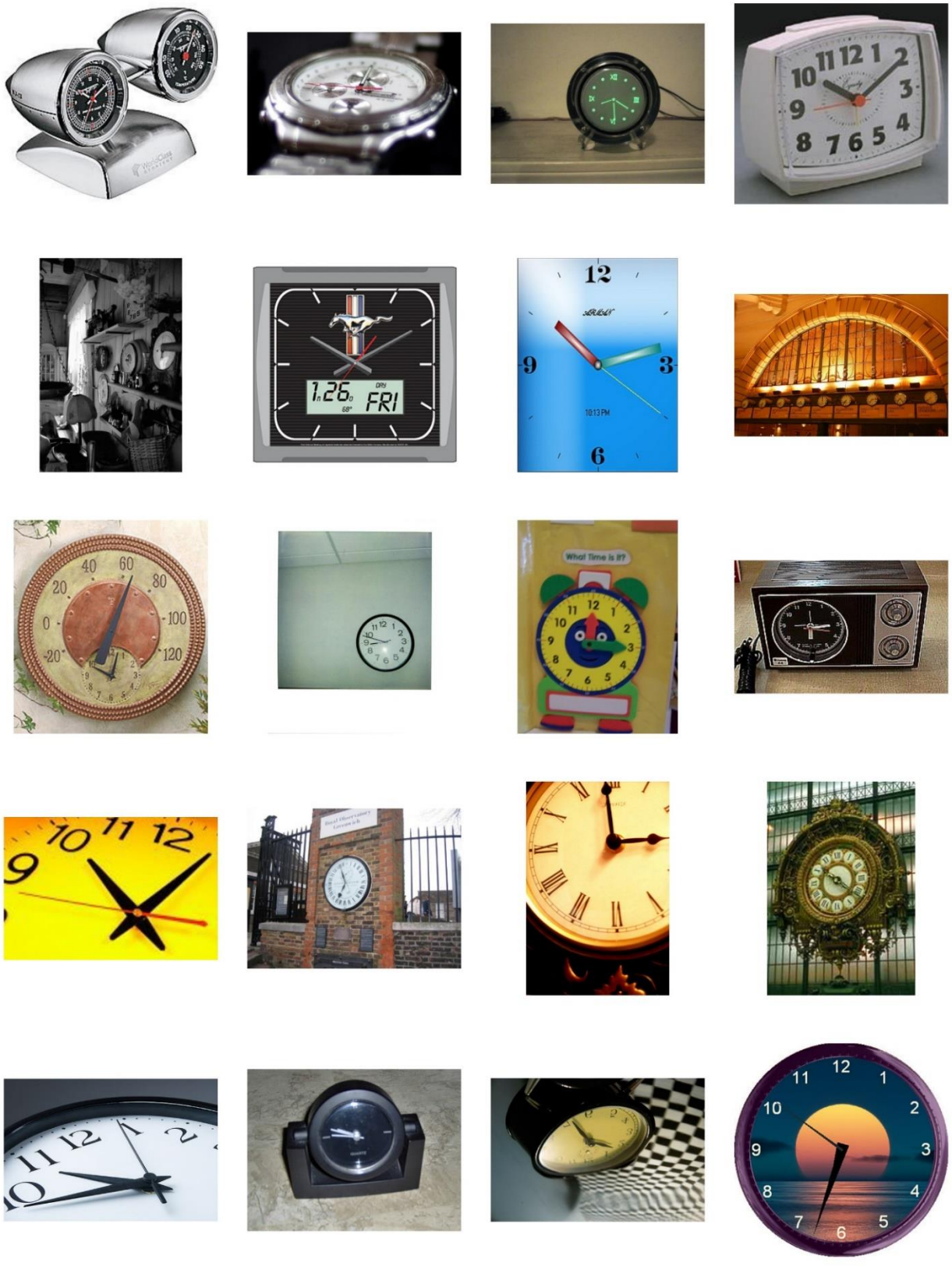

Figure 10 Intra class variance: ImageNet alarm clock

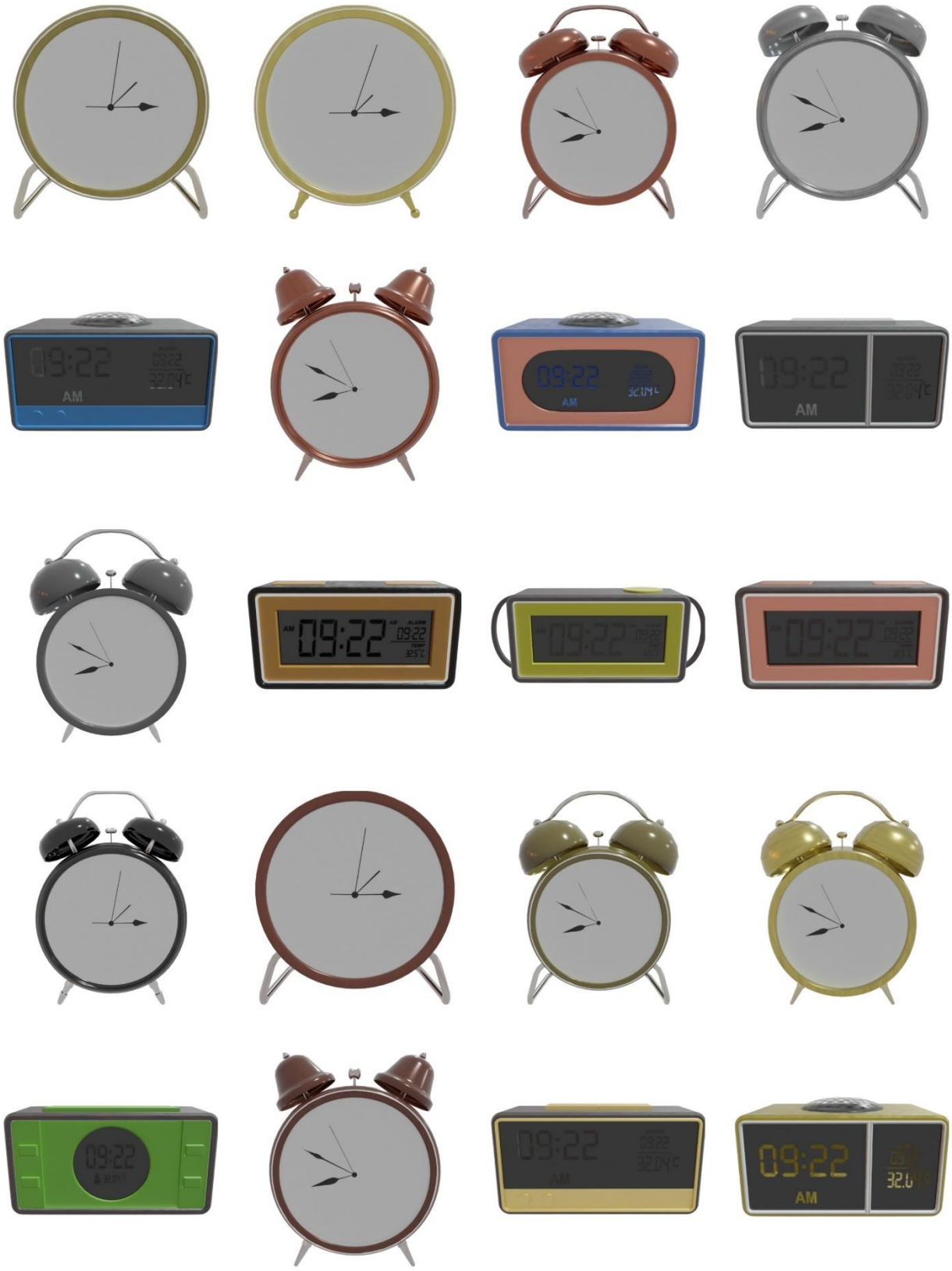

Figure 11 Intra class variance: synthetic alarm clock

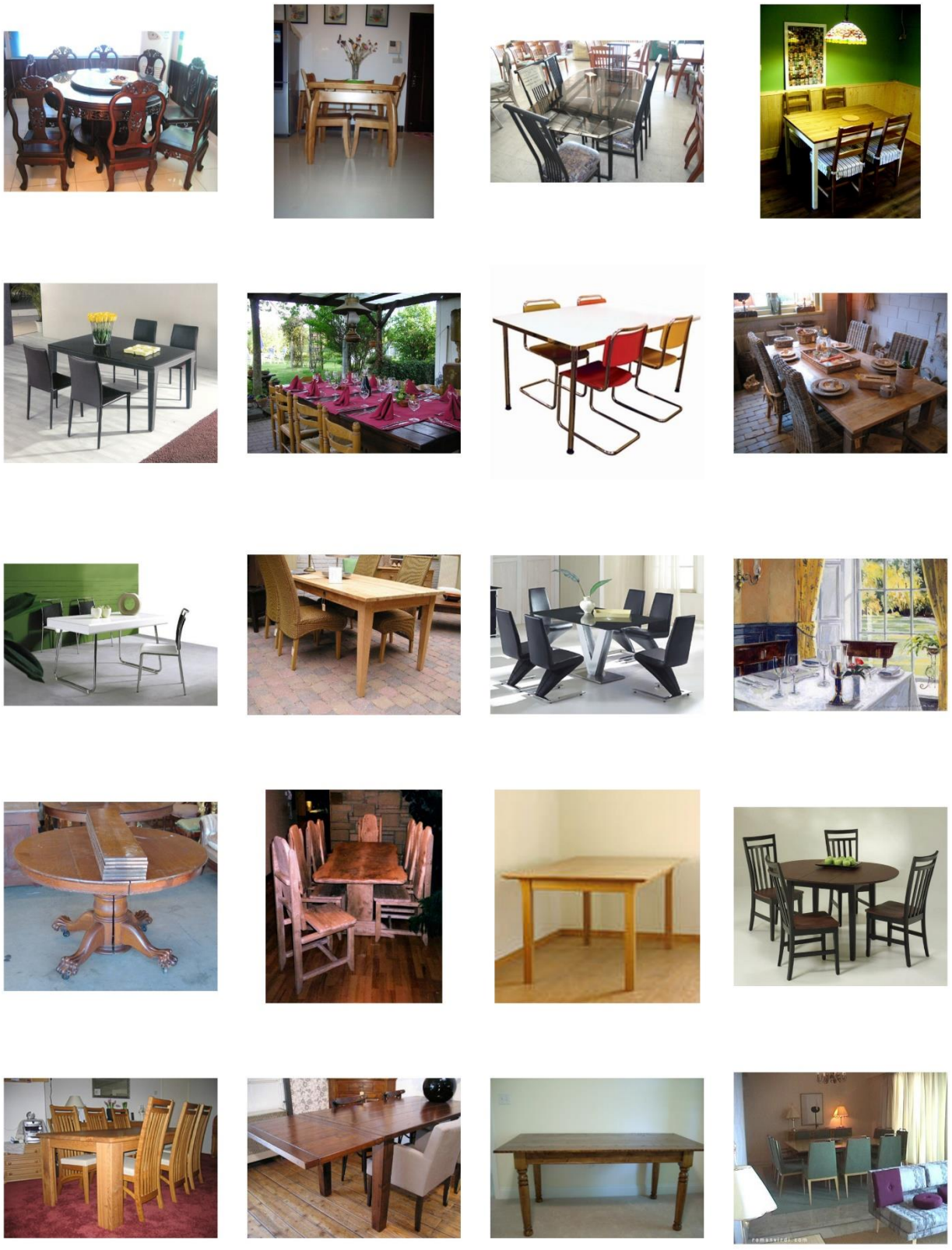

Figure 12 Intra class variance: ImageNet food table

Figure 13 Intra class variance: synthetic food table

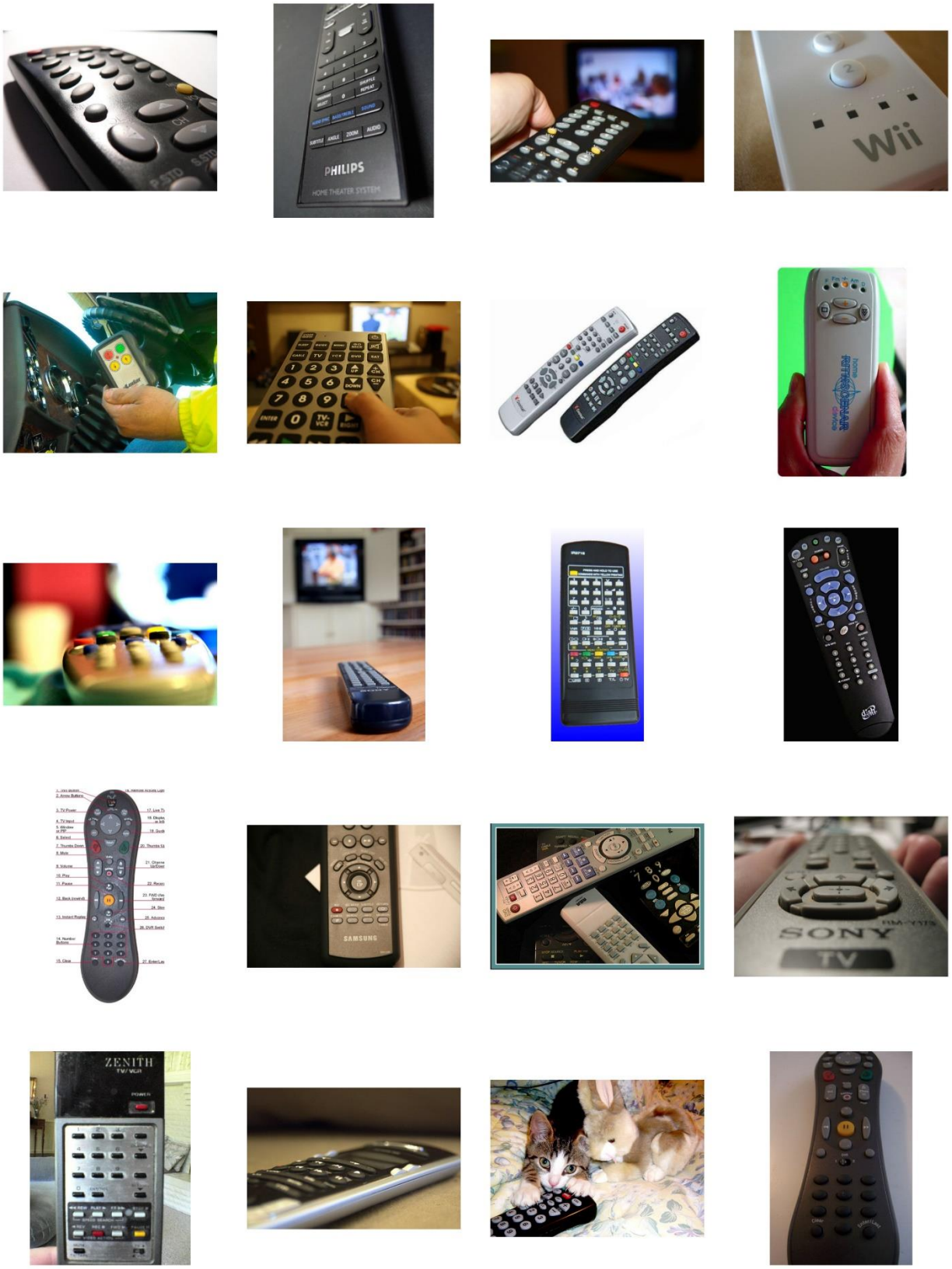

Figure 14 Intra class variance: ImageNet tv remote

Figure 15 Intra class variance: synthetic tv remote

5. Methodology

5.1. Synthetic Data Generation

For the synthetic data generation a proprietary high quality professionally designed 3D models were used. As a control 3D software, an open source Blender [5] has been utilized. Blender enables photorealistic rendering and supports Python API, enabling algorithmic control.

Object image generation

The following pseudocode has been implemented:

1) For each Object variation from the Classes Of Interest:
    a. For each angle {grid of angles} over a single Object in the frame
        i. Set the camera at a desired view angle
        ii. Set the camera so that an Object span in frame is maximal
        iii. Set transparent background for the rendering
        iv. Render the image
        v. Save the image

Merging algorithm

The object image and the background image composition is achieved by a simple algorithm. The algorithm takes as input 2 randomly picked images – object image (foreground) and a background image. It assumes that 1) the object image includes only the object itself located on a transparent background 2) the foreground image is bigger than the background image. The algorithm first randomly rotates the foreground image within uniform distribution [0°, 360°). It calculates the maximum range of foreground image resize by extracting the minimum of proportions of x and y coordinates – in order to get the foreground image to be merged completely into the background image. The resize value is calculated by randomly (uniform) choosing a value from the range [maximum range * 0.5, maximum range]. Finally, the foreground and the background images are merged by pasting the foreground into the background.

5.2. Deep Learning Model

Implementation

The deep learning implementation is based on the official PyTorch object classification implementation customized for the ImageNet dataset. The actual implementation is taken from https://github.com/pytorch/examples/tree/master/imagenet.

Augmentations

Augmentations for the real dataset and the synthetic dataset are different.

For the real data, the transformations used during the training: RandomResizedCrop(224), RandomHorizontalFlip(), normalize. Transformations used during validation: Resize(256), CenterCrop(224), normalize.

For the synthetic data: Resize([224, 224]). With augmentation mode added: RandomAffine(degrees=(0,180), translate=(1/3,1/3), scale=(0.1,1.0)). Augmentation mode was enabled during with a background experiments.

Training hyper parameters

The initial hyper parameters were taken from the official implementation. Further in the experiments different sets of parameters have been tested.

Architecture	ResNet 50
Learning rate	0.01, decays by a factor of 10 every 30 epochs
Epochs	120
Batch size	64

Table 9 Initial hyper parameters

Metrics

The following metrics were used: accuracy top 1, 5, additional top 3, top 1 precision and top 1 recall. The additional top 3 was added to keep fair results after the number of classes was reduced from 1000 down to 17. The probability for the network to be correct in top 5 predictions is much higher with 17 classes instead of 1000. In addition, top 1 accuracy has been measured per each object class.

Pre train

In all experiments no pre trained network has been used – it was always trained with initially random parameters.

5.3. Experiments

5.3.1. Baseline

The baseline represents the reference model, trained on a real dataset. Specifically, a deep learning network trained on the selected 17 classes from an ImageNet dataset. The full 1,000 object classes ImageNet dataset has been reduced to a subset of 17 object classes. The dataset images were exactly as the original without any modifications. To measure the baseline benchmark, the network implementation and augmentations, remained as the original. See Table

11. The baseline is marked with a blue color in all the experiments presented in the Results section.

5.3.2.Real Train, Synthetic Test

The baseline trained model has been tested on the synthetic target test datasets. It was tested on both synthetic with and without any background datasets. The goal was to measure the domain gap from real to synthetic direction. See Table 11.

5.3.3.Synthetic without any Background

Data centric experiments. In this experiments set the training data was a set of object images only – no image backgrounds merged. The goal was to analyze the effect of adding a background. On one hand, an object image with a background is expected to minimize the domain gap by turning the images closer to the real world representation. On the other hand, without any background, an object image holds the focused information of the object itself, without additional noise the background adds, turning it to a potentially cleaner signal to the network.

The models were tested on both real and synthetic datasets. Real test dataset was the ImageNet validation dataset, the same as the baseline was benchmarked. The synthetic test dataset was created by sampling total 20% images from the full dataset. See Table 12, Table 14.

5.3.4.Synthetic with a Background

Data centric experiments. In this experiments set a merging algorithm was applied over each training datapoint. The object image was merged with a random background adding augmentations to the object image. It was implemented to be merged online during the network training just before the data insertion into the network's input layers. The models were tested on both real and synthetic datasets. See Table 14.

5.3.5.Synthetic with Various Hyperparameters

Code centric experiments. All the experiments were trained with a synthetic with a background dataset. The basic implementation remained the same, with different sets of hyperparameters. The goal was to check how hyperparameters impact the performance and find whether there's hyperparameters set that might significantly boost the performance. Test dataset included the real data only samples.

In particular the following hyperparameters were tested. Resnet 34, 50, 101 network architecture. SGD and Adam optimizer algorithm. 0.01 and 3e-4 initial learning rates. Learning rate auto adjustment mode was tested for both on and off. See Table 18

5.3.6.Mix of Synthetic and Real

Data centric experiments. The training dataset is a mixture of both synthetic and real data. The mixture was done with different proportions with a goal to analyze both synthetic and real data contributions.

Real/Synthetic contribution
2 main sets of experiments have been performed. First set, analysis of the synthetic data contribution, where a full real dataset and varying sizes of synthetic participated in the training. In particular 19,854(10%), 70,047(37%), and 187,928(100%) of the available synthetic dataset and 1,200 ImageNet images in all the experiments. Second set, analysis of real data contribution, where a full synthetic and varying sizes of real data participated in the training. In particular 40*17(3.3%), 200*17(16%), 400*17(33%), 1,200*17(100%) of the ImageNet and 187,928 synthetic images in all the experiments within this set. Test dataset was the real in all the experiments.

Results reliability
The goal of these experiments was to verify the reliability of the achieved results by analyzing the statistical stability.

The reproduced experiment was the full synthetic dataset mixed with 400*17(33% of the full dataset) ImageNet as a training data. This experiment configuration was intentionally selected. As the results show in Table 23, a higher ratios (>33%) of real data, produce a better performance compared to the baseline, while lower ratios (<33%) achieve a worse performance. Hence the 33% ratio is the balance point, where approximately the same performance as the baseline is achieved (when 100% of the ImageNet dataset participated).

The full training ImageNet dataset was split into 3 mutually exclusive sets with 400 images per class (total of 6,800 images). In addition, we added the full synthetic training set to each of those 3 sets. For each training set, we created 5 test sets sampled from the remaining training samples, with a 0 intersection between {test set1, …, test set5}. Size of each test set was 180 images per class (3,060 total). Test set was a real data only from the ImageNet samples. Then 3 train sessions were executed over each of the 3 train datasets. And eventually, each trained model tested against the corresponding 5 test datasets. Producing total of 3x5=15 test results. See Table 24.

6. Results

6.1. Baseline

Index	Train	Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
1	Imagenet	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	66.82	88.12	94.35	68.86	66.82
2	Imagenet	Imagenet	Resnet 50 Optimizer=SGD LR=0.1 LRadjust=True	62.00	86.47	91.88	63.34	62.00

Table 10 the baseline model

Table 10 demonstrates the results obtained when training and testing the model on ImageNet subset Table 2. The model was trained and tested with initial learning rate settings of 0.01 and 0.1. According to the official implementation, the recommended learning rate for ResNet 50 and the full 1,000 samples per class ImageNet is 0.1. However, 0.01 produced a better results than 0.1, therefore this version will serve as the baseline in this work.

6.2. Real Train, Synthetic Test

Index	Train	Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
1	Imagenet	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	66.82	88.12	94.35	68.86	66.82
2	Imagenet	Synth w/ BG	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	21.91	43.63	57.80	18.86	22.27
3	Imagenet	Synth wo/ BG	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	33.52	53.49	66.81	29.62	31.86
4	Imagenet	Imagenet 400 samples	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	59.65	83.18	91.18	45.92	43.76

Table 11 Results: trained on real, tested on synthetic

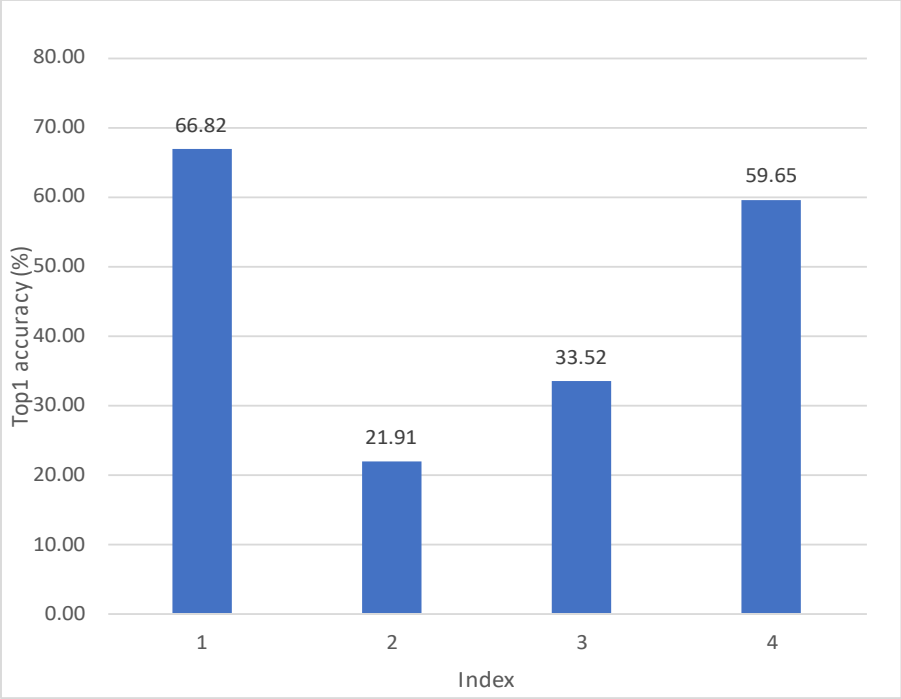

Figure 16 Results: trained on real, tested on synthetic

It is evident that the performance degrades significantly compared to the baseline (marked with blue text), in case of a model trained on ImageNet and tested on synthetic datasets. Based on a relatively good results of the baseline performance, where the model infers on a never seen real data, it suggests the trained model is robust enough, at least within the same domain – for the real test data. Therefore the performance degradation in case when the test data is synthetic, is most likely described by the domain gap.

It is interesting to note that the real data trained model inference on a without any background dataset produces a better results than with a background dataset. Without any background includes the object image only, therefore it is much cleaner, noise free (considering object classification task) and object only focused. So it suggests the background noise makes it harder for the model to predict correctly.

6.2.1.Synthetic Train, Synthetic Test

Index	Train	Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
1	Synth with BG	Synth with BG	Resnet 50 Optimizer=SGD LRi=0.1 LRadjust=True	97.01	99.58	99.84	96.35	96.14
2	Synth without BG	Synth without BG	Resnet 50 Optimizer=SGD LRi=0.1 LRadjust=True	99.46	100.00	100.00	99.02	98.89

Table 12 Results: synthetic trained, tested on synthetic same configuration

The same configuration experiments – where training and test data are from the same configuration in aspect of the background containment. The results demonstrate a strong performance in both configurations. This indicates that the network does learn from the training data.

Index	Train	Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
1	Synth with BG	Synth without BG	Resnet 50 Optimizer=SGD LRi=0.1 LRadjust=True	99.17	99.95	99.97	98.59	98.35
2	Synth without BG	Synth with BG	Resnet 50 Optimizer=SGD LRi=0.1 LRadjust=True	7.84	18.66	28.87	16.88	10.62

Table 13 Results: synthetic trained, tested on synthetic cross category

In different configuration experiments – where training and test data are from a different configuration in aspect of the background containment. The results show a strong performance in configuration with background trained data and tested on without any background. This leads to a similar conclusion as in the case of a real trained, tested on without any background case Table 11, it is easier for the model to handle noise free images, after trained on a harder, noisy, samples. The performance degrades significantly in the exact opposite case – trained on without any background, tested on with a background.

From the results we can conclude that training with background yields a much more robust model turning it a preferred configuration option.

6.3. *Synthetic Train, Real Test*

Index	Train	Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
1	Imagenet	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	66.82	88.12	94.35	68.86	66.82
2	Synth with BG	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	16.00	32.71	45.18	18.48	16.00
3	Synth without BG	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	8.82	22.24	34.12	8.23	8.82
4	Synth with BG (1200 per class)	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	11.88	26.47	40.59	16.01	11.88

Table 14 Result: trained on synthetic, tested on real (table)

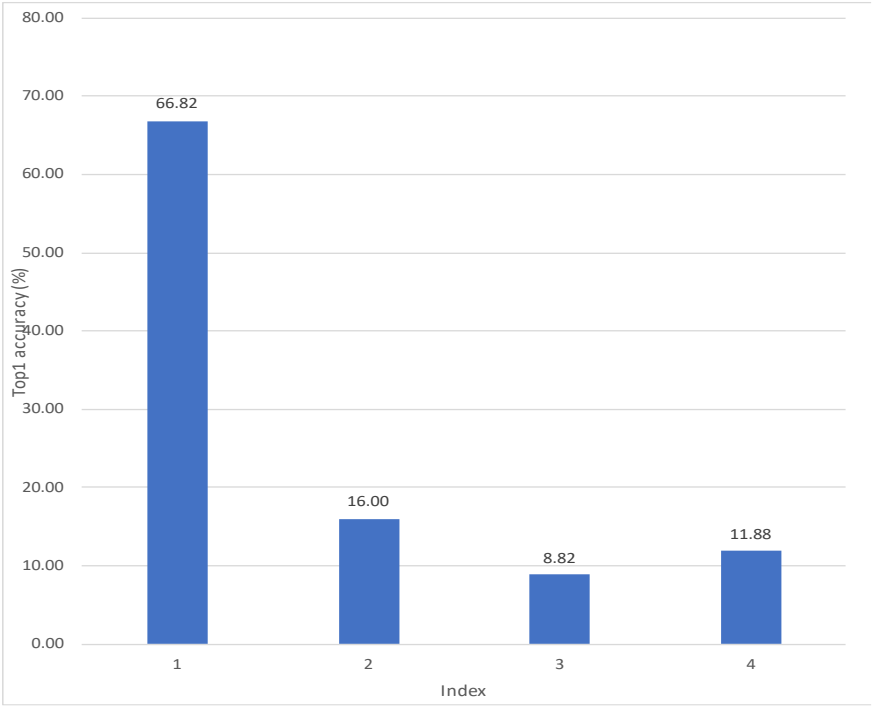

Figure 17 Results: trained on synthetic, tested on real (figure):

The results show how the performance degrades significantly, compared to the baseline, when testing a synthetic dataset trained model on a real data. The domain gap is symmetric – does not

depend on the train-test direction real → synthetic or synthetic → real. As shown previously (see Table 11), in the real → synthetic case there was a domain gap. Therefore it is expected to exists also for the synthetic → real direction. This partially explains the performance degradation.

Similarly to the results in Table 13, trained on synthetic with background model performs better than without any background when tested on a real data. It again demonstrates that with a background synthetic data produces a more robust model, and specifically approximates better the real data.

The results in Table 14 index 5 demonstrate a model trained on synthetic data with the exact same quantities as in ImageNet – 1200 samples per class. This dataset has been prepared by sampling concrete viewpoints along with some random ones from the full dataset, constrained to include all the instances appearing in the full dataset. As expected, the results demonstrate degradation compared to the configuration when trained on the full synthetic dataset (index 2 in Table 14).

alarm_clock	4%
backpack	22%
banana	12%
basketball	0%
computer_desk	44%
cup	2%
digital_clock	2%
food_table	68%
laptop	0%
pc_keyboard	14%
pc_monitor	8%
pc_mouse	12%
soccer	28%
toaster	0%
tv_remote	28%
vase	10%
wall_clock	18%

Table 15 per class top-1 accuracy with a background: full synthetic dataset

alarm_clock	4%
backpack	4%
banana	2%
basketball	0%
computer_desk	0%
cup	2%
digital_clock	2%
food_table	0%
laptop	28%
pc_keyboard	4%
pc_monitor	0%
pc_mouse	12%
soccer	42%
toaster	38%
tv_remote	0%
vase	8%
wall_clock	4%

Table 16 per class top-1 accuracy without any background

alarm_clock	0%
backpack	12%
banana	8%
basketball	2%
computer_desk	60%
cup	4%
digital_clock	0%
food_table	48%
laptop	8%
pc_keyboard	0%
pc_monitor	10%
pc_mouse	10%
soccer	26%
toaster	0%
tv_remote	4%
vase	4%
wall_clock	6%

Table 17 per class per class top-1 accuracy with a background: 1200 per class synthetic dataset

As Table 15, Table 16 show some classes perform better with a background and some without any background configurations when tested on ImageNet. Synth hybrid BG (best of the both configurations per class), that maximizes the performance of the both configurations did not produce any improved results.

Table 17 shows the experiment where synthetic data has been trained with 1200 samples per class. The final performance is lower than the full synthetic dataset, but the performance patterns per class are similar. For example food table and computer desk produce high results, and toaster, basketball and digital clock produce low results in both cases. The confusion matrix Figure 18, Figure 19 demonstrate a similar patterns. Therefore, the bigger the synthetic dataset the more it improves the performance for each class.

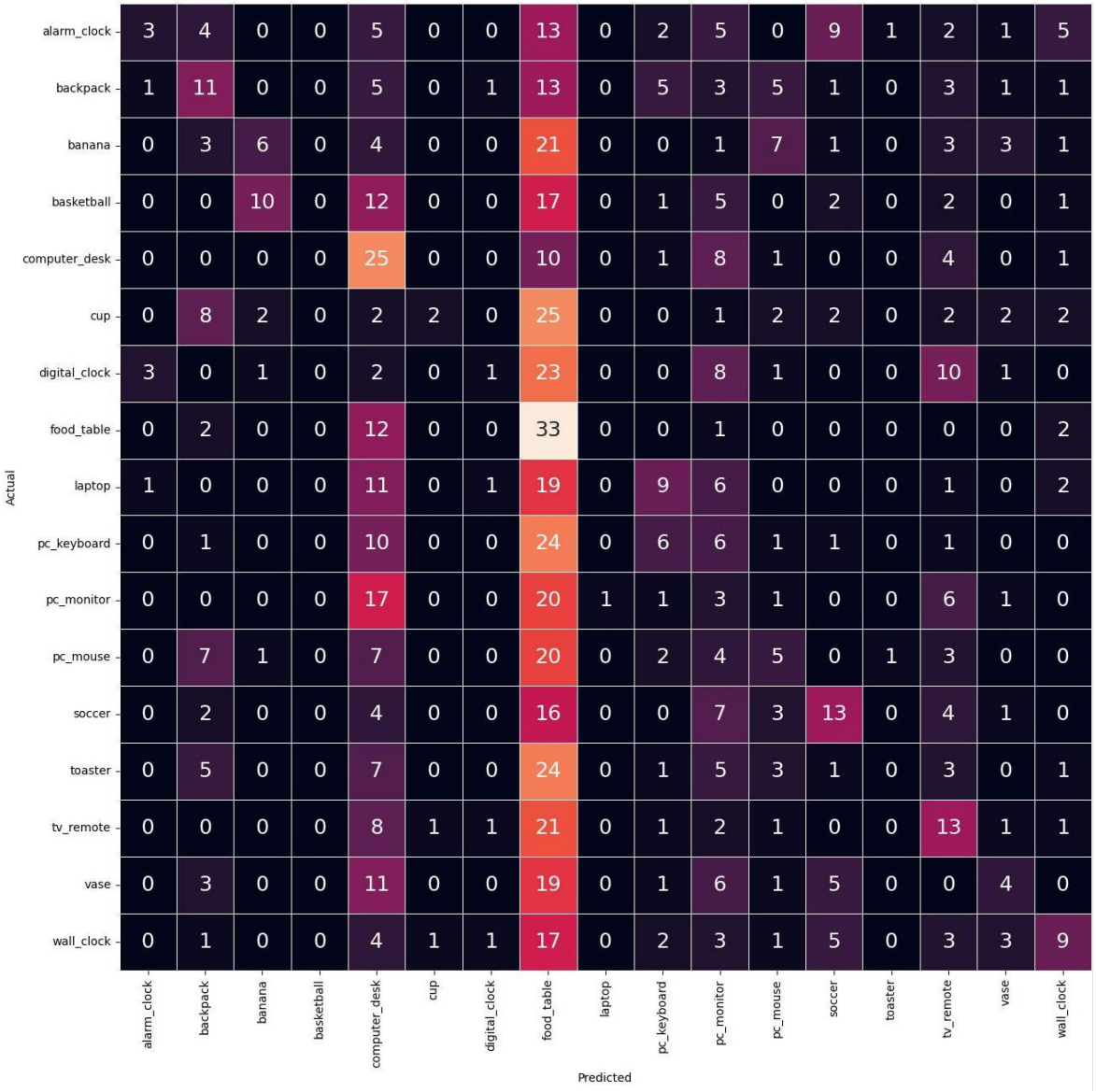

Figure 18 per class confusion matrix: full synthetic dataset

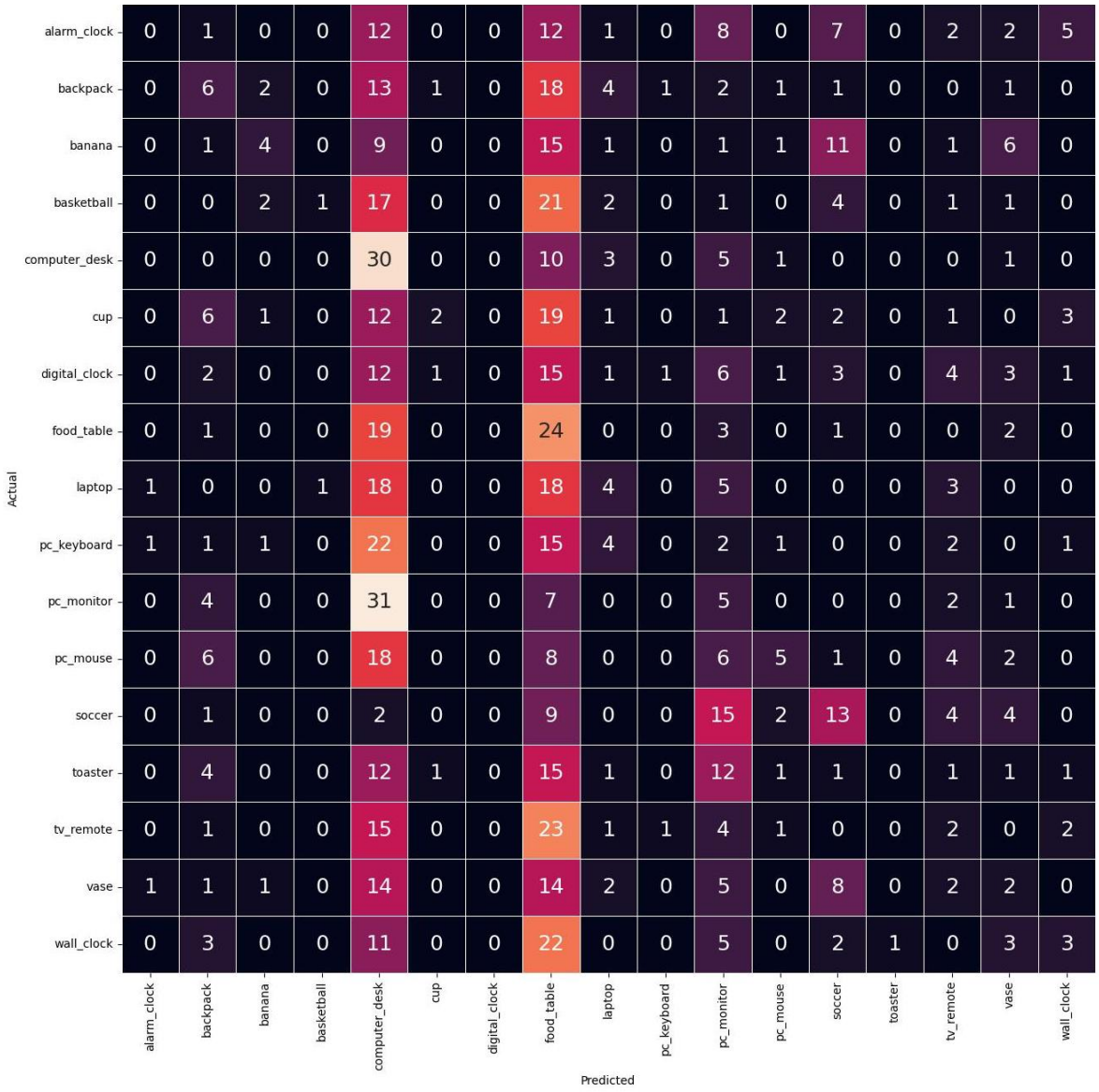

Figure 19 per class confusion matrix: 1200 per class synthetic dataset

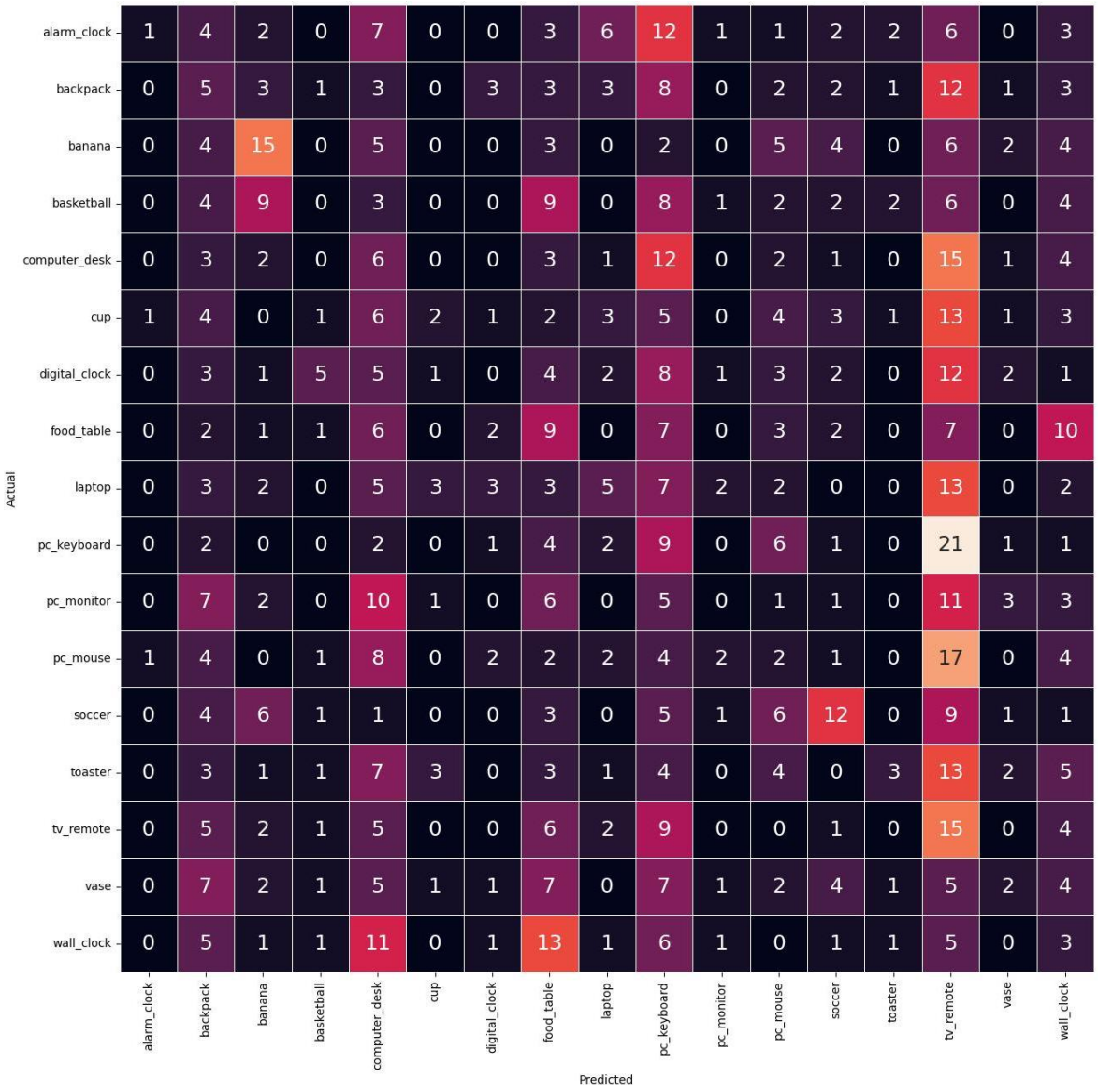

Figure 20 per class confusion matrix: 1200 per class synthetic dataset - non sparse food table and computer desk

Figure 21 food table (top), computer desk (bottom) sparse examples

Figure 22 food table (bottom), computer desk (top) non sparse examples

Figure 18 represents the confusion matrix when the model was trained on the full synthetic dataset. The matrix shows that most of the model predictions are food table and computer desk classes. Since the model predicts most of the time those classes, it explains the high accuracy score in Table 15 produced for those 2 classes, but it comes with a lots of false positive results. It looks like that the model is strongly biased towards those 2 classes. The same pattern might be observed also for the balanced dataset version in Figure 19. If we analyze some examples (see Figure 21) used for training to produce Figure 19 results. The common characteristics between the classes is that the objects cover a relatively small image area (sparse image coverage), leaving a major area for the background image, therefore, apparently the model mostly learns the backgrounds labeled as those classes. Since some of the backgrounds are shared between the classes across the dataset (see the mapping Table 3), the same background might appear for both food table and alarm clock instances, during the training. To test this connection, we replaced the sparse viewpoints Figure 21 with a non-sparse "from the top" viewpoints Figure 22 for food table and computer desk in the same training data set as before, and trained the model again. Such viewpoints cover more of the target objects, leaving less space for the background, hence making them less sparse. This time the results Figure 20 show the bias has been drastically reduced for those food table and computer desk classes. However, this time they present for tv remote and pc keyboard. And the final results are similar to the previous dataset version.

6.4. Synthetic with Various Hyperparameters

Index	Train	Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
1	Synth with BG	Imagenet	Resnet 34 Optimizer=SGD LRi=0.01 LRadjust=True	**19.65**	36.12	48.59	27.11	19.65
2	Synth with BG	Imagenet	Resnet 101 Optimizer=SGD LRi=0.01 LRadjust=True	19.06	38.71	**51.41**	21.32	19.06
3	Synth with BG	Imagenet	Resnet 50 Optimizer=SGD LRi=0.1 LRadjust=True	16.12	36.24	50.35	27.02	16.12
4	Synth with BG	Imagenet	Resnet 50 Optimizer=Adam LRi=3e-4 LRadjust=True	17.76	36.94	50.47	28.42	17.76
5	Synth with BG	Imagenet	Resnet 50 Optimizer=SGD LRi=0.01 LRadjust=False	17.29	39.53	53.53	34.66	17.29
6	Synth with BG	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	16.00	32.71	45.18	18.48	16.00

Table 18 Results: Hyperparameters tuning

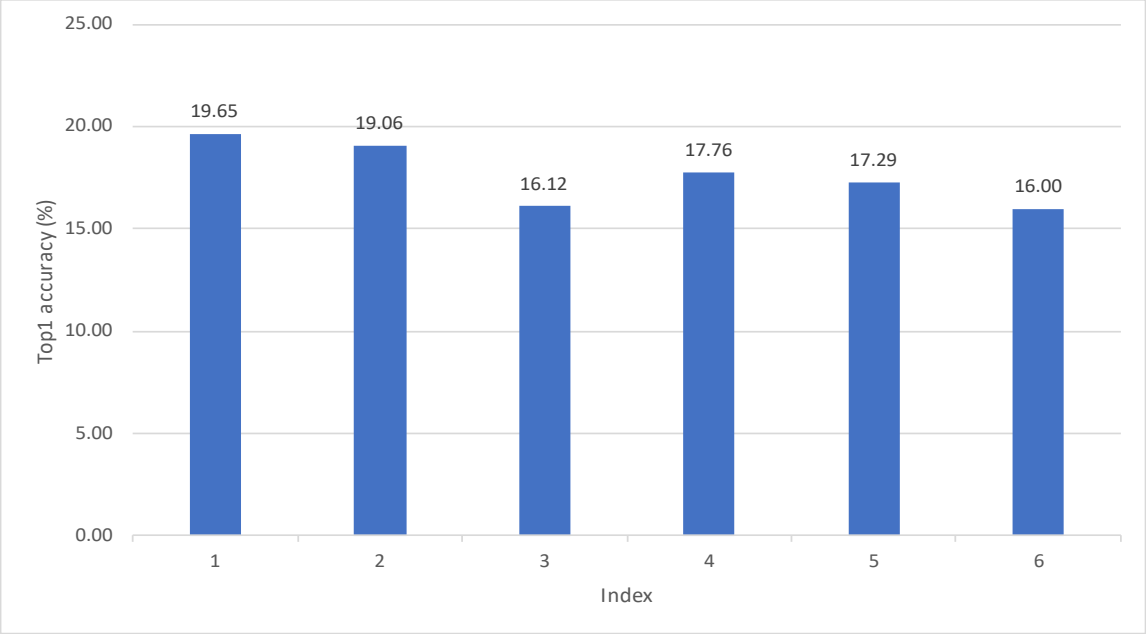

Figure 23 Results: Hyperparameters tuning

The baseline hyperparameters configuration is marked with an orange color text. Some result, under different hyperparameters, show minor improvements in the performance compared to the baseline configuration. However, no desired performance gain has been achieved. Interesting to note that bigger and smaller networks: Resnet 34 and Resnet 101 perform better than the baseline's Resnet 50.

For conclusion, hyperparameters tuning produced no significant change.

6.5. *Mix of Synthetic and Real*

Index	Train		Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
	Real #(%)	Synthetic #(%)							
1	20,400(100%)	0%	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	66.82	88.12	94.35	68.86	66.82
2	20,400(100%)	187,928(100%)	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	**75.65**	**93.29**	**97.29**	**76.61**	**75.65**
3	20,400(100%)	70,047(37%)	Imagenet	---	73.06	91.65	96.47	73.94	73.06
4	20,400(100%)	19,854(10%)	Imagenet	---	70.59	89.76	94.71	72.54	70.59
5	20,400(100%)	187,928(100%)	Imagenet	Resnet 101 Optimizer=SGD LR=0.01 LRadjust=True	77.18	93.41	97.41	78.04	77.18

Table 19 Results: Contribution of synthetic data

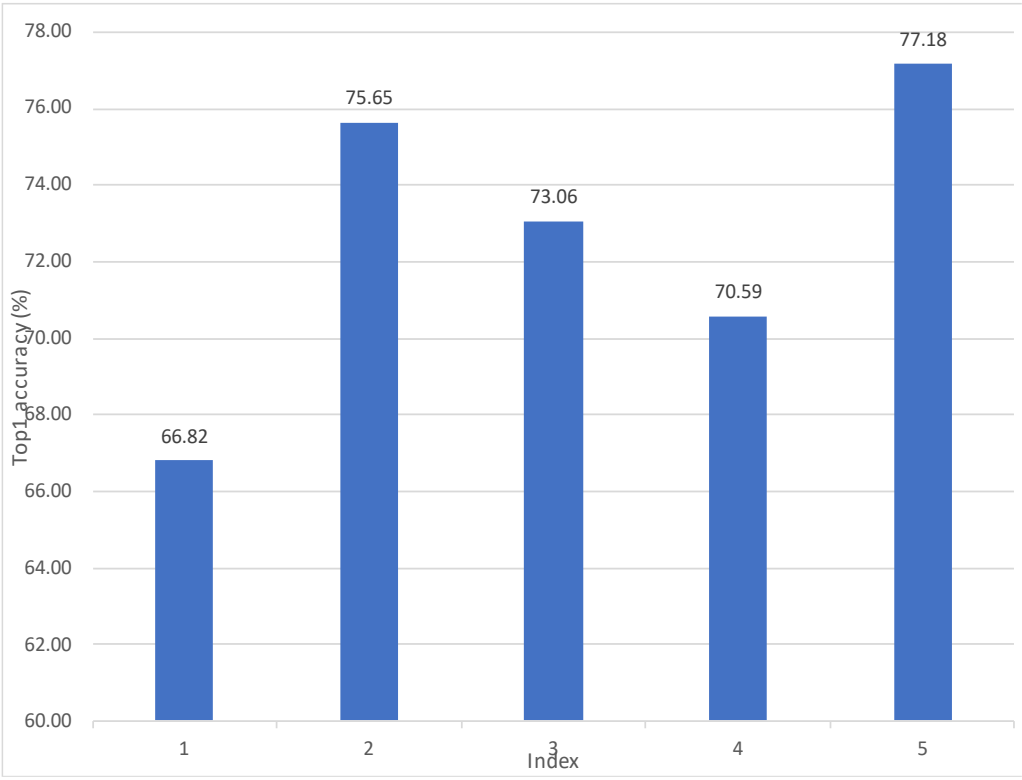

Figure 24 Results: Contribution of synthetic data

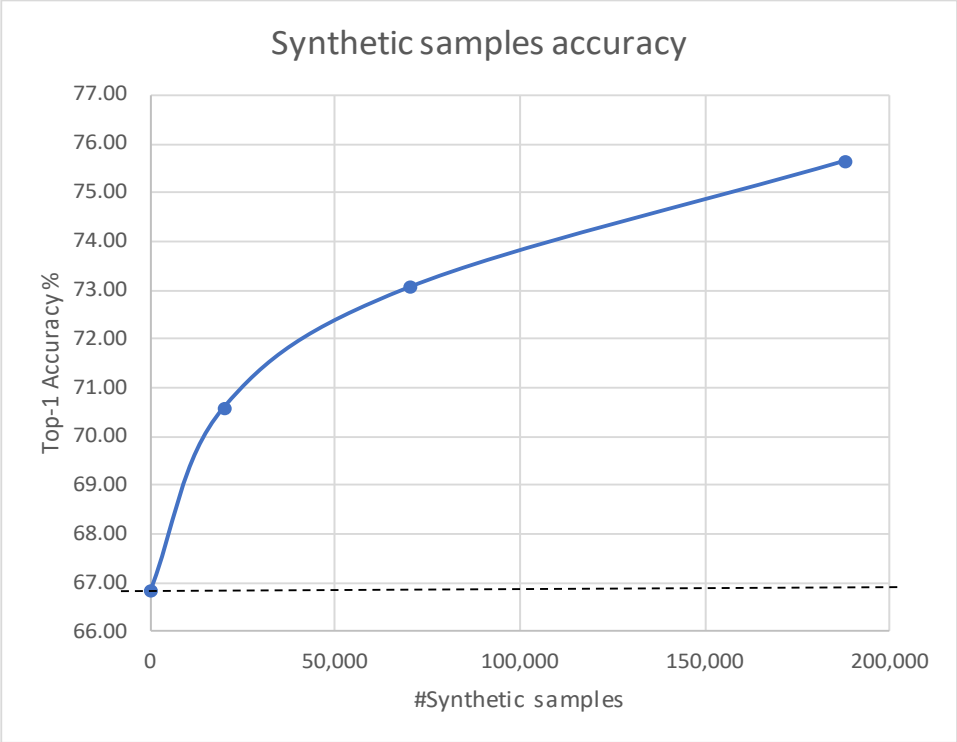

Figure 25 Summary: Contribution of synthetic data. Dashed line represents the baseline

The results demonstrate that adding synthetic data to the real data, led to a performance gain. And the bigger the addition, the more the gain. When trained with both synthetic and real datasets compared to the baseline we achieve a 13.16% performance gain in top-1 accuracy for the same network architecture and hyperparameters Table 19 (index 1 and index 2). That being said, it is also apparent the performance gain is minor on each synthetic data addition with big values. Figure 25 represents the synthetic data size and the performance gain correlation. For example an addition that almost doubles the training dataset of 19,854 synthetic images to the full 20,400 images ImageNet yields only additional 3.77% in the performance boost, compared to the baseline. There's an additional 1.53% gain achieved with the Resnet 101 compared to Resnet 50.

For conclusion, the synthetic data contributes to the performance, but a relatively minor contribution taking into account the quantities involved.

alarm_clock	44%
backpack	58%
banana	72%
basketball	82%
computer_desk	74%
cup	64%
digital_clock	66%
food_table	84%
laptop	70%
pc_keyboard	72%
pc_monitor	52%
pc_mouse	46%
soccer	90%
toaster	70%
tv_remote	60%
vase	58%
wall_clock	74%

Table 20 per class top-1 accuracy: baseline real(100%)

alarm_clock	62%
backpack	64%
banana	92%
basketball	90%
computer_desk	76%
cup	80%
digital_clock	76%
food_table	94%
laptop	72%
pc_keyboard	74%
pc_monitor	54%
pc_mouse	52%
soccer	98%
toaster	84%
tv_remote	82%
vase	66%
wall_clock	70%

Table 21 per class top-1 accuracy mixed: synthetic(100%)+real(100%)

alarm_clock	18
backpack	6
banana	20
basketball	8
computer_desk	2
cup	16
digital_clock	10
food_table	10
laptop	2
pc_keyboard	2
pc_monitor	2
pc_mouse	6
soccer	8
toaster	14
tv_remote	22
vase	8
wall_clock	-4

Table 22 difference between baseline and mixed

Analyzing how the mixed training mode contributes to each class performance gain, as shown in Table 22, shows the gain difference for each class. In fact, there's even a class – wall clock where the performance degrades a bit. But all other classes perform better up to additional 22% compared to the baseline.

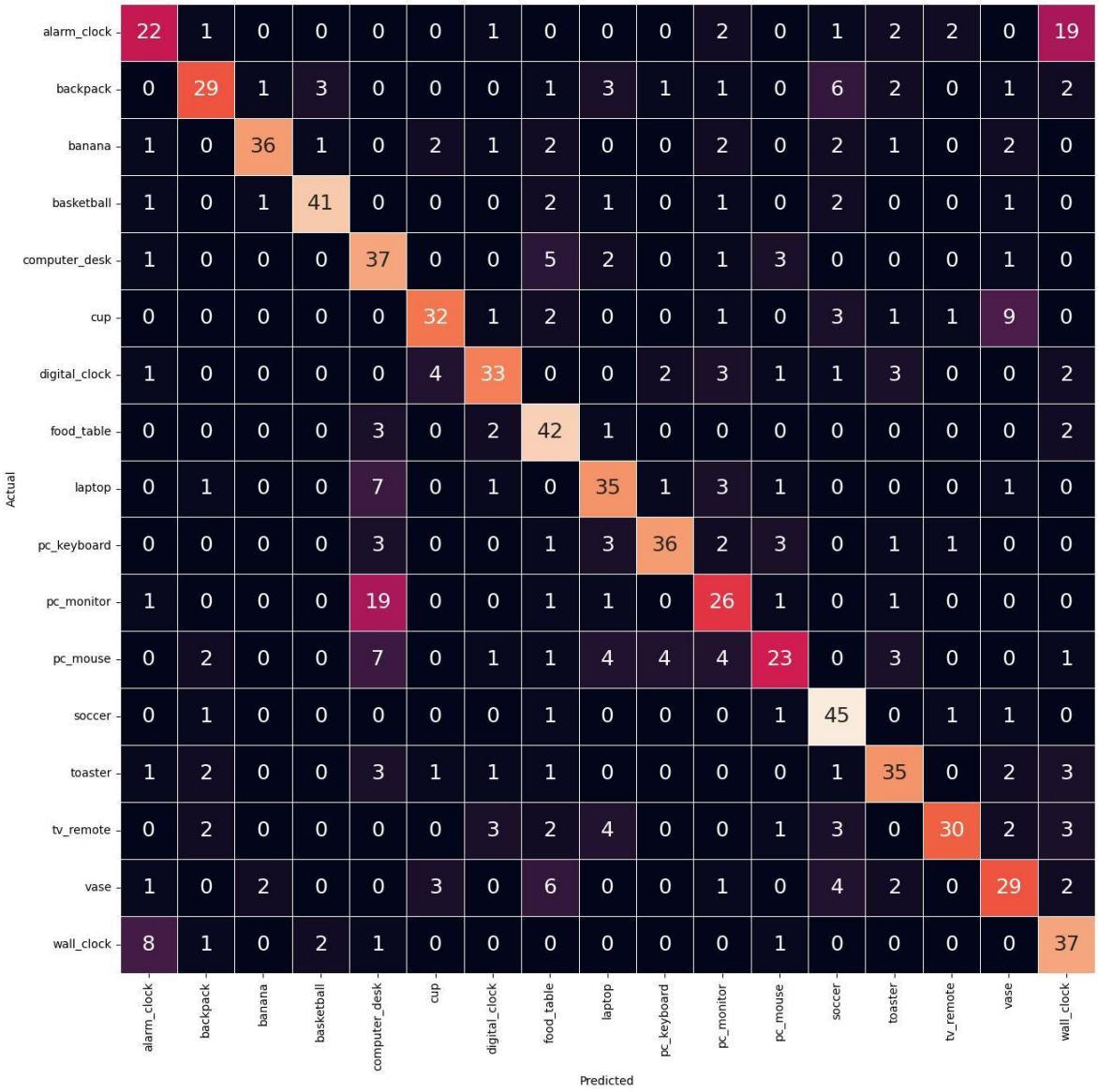

Figure 26 per class confusion matrix: baseline real(100%)

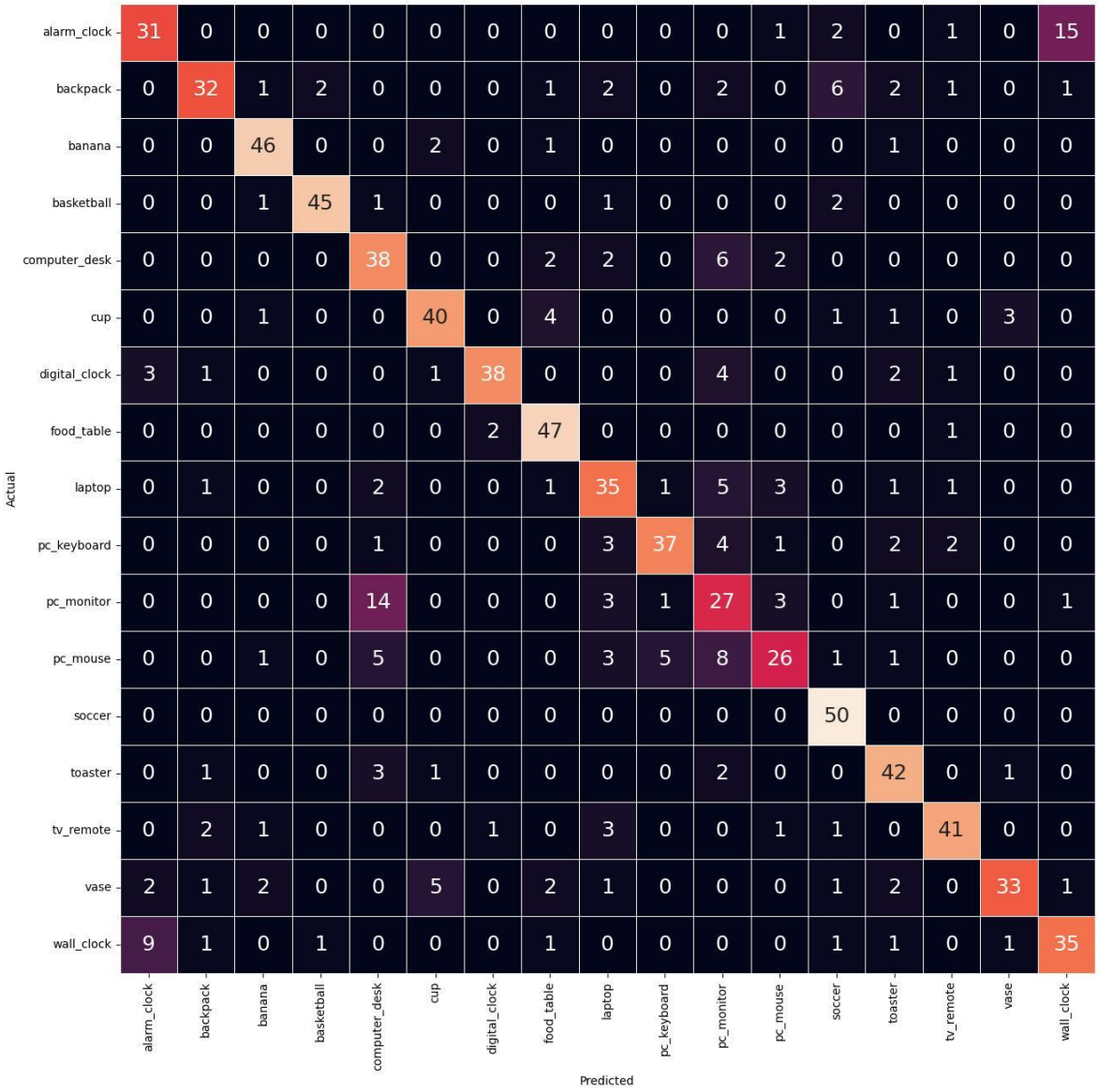

Figure 27 per class confusion matrix mixed: synthetic(100%)+real(100%)

Figure 26, Figure 27 and Figure 32 show the confusion matrix of the classified samples per each class (50 samples per class). Not surprisingly, the models in both cases Figure 26, Figure 27 confuse between alarm and wall clock. The samples from the both classes look very similar. There are also some misclassifications between PC monitor and computer desk, since the monitors are usually on computer desk in the images, no surprise here as well. More errors originate mainly from misclassification of computer devices.

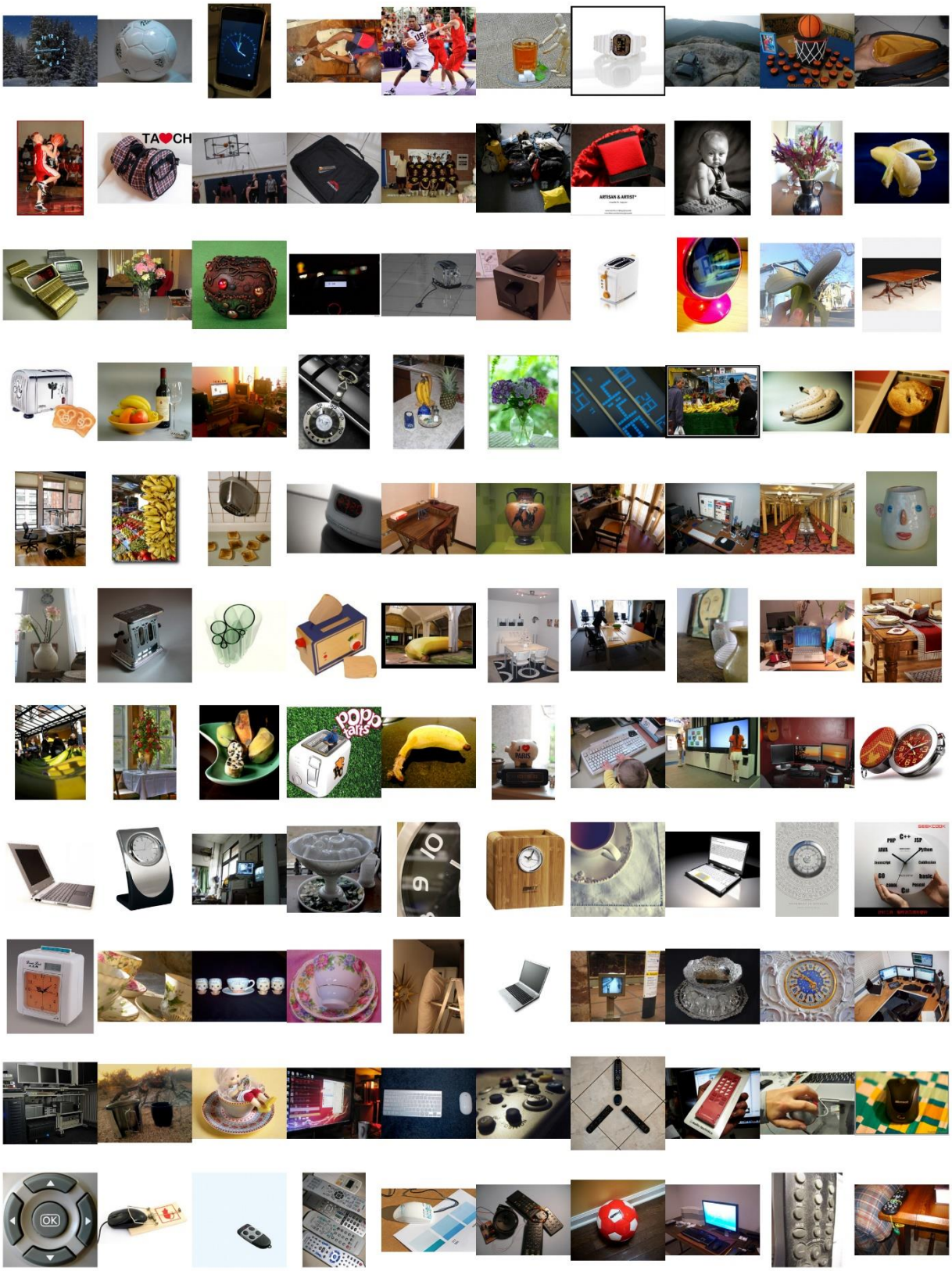

Figure 28 Correctly classified samples contributed by synthetic data

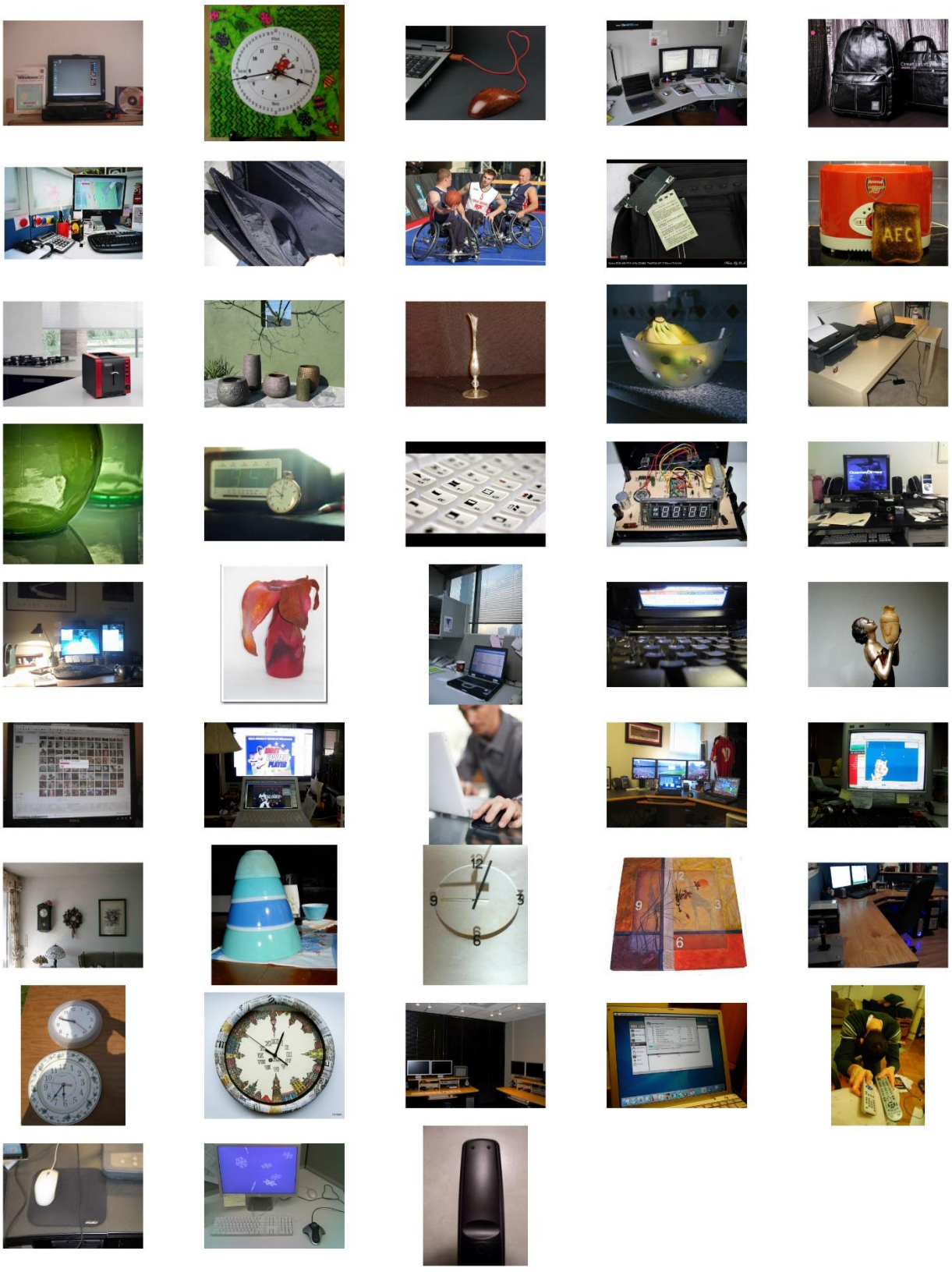

Figure 29 Misclassified samples contributed by synthetic data

In order to analyze how the synthetic data contributes to the model knowledge, we compare the model trained on 100% real and 100% synthetic datasets Table 19 (index 3) against the 100% real baseline model (index 1). In both cases the test set is the same (real dataset) and consists of 850 samples. In case of the index 3 model, there were 643 correctly classified samples and 207 incorrectly. The baseline classified 568 correctly and 282 incorrectly. Number of the same samples both of the models classified correctly was 525 and 164 incorrectly. The index 3 model learnt to properly classify an additional 118 samples (compared to 1). See Figure 28 for the concrete samples. But in the same time, index 3 model misclassified 43 samples, that the baseline classified correctly. See Figure 29 for the concrete samples. Therefore, the addition of the synthetic data to the training dataset helped the model perform better for some samples cases, but also perform worse for some others.

Index		Train	Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
	Real #(%)	Synthetic #(%)							
1	20,400(100%)	0%	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	66.82	88.12	94.35	68.86	66.82
2	20,400(100%)	187,928(100%)	Imagenet	---	**75.65**	**93.29**	**97.29**	**76.61**	**75.65**
3	6,800(33%)	187,928(100%)	Imagenet	---	65.41	88.82	94.47	66.17	65.41
4	3,400(16%)	187,928(100%)	Imagenet	---	59.65	83.18	91.18	60.04	59.65
5	680(3.3%)	187,928(100%)	Imagenet	---	49.61	74.61	82.03	43.98	43.88
6	0(0%)	187,928(100%)	Imagenet	---	16.00	32.71	45.18	18.48	16.00

Table 23 Results: Contribution of real data

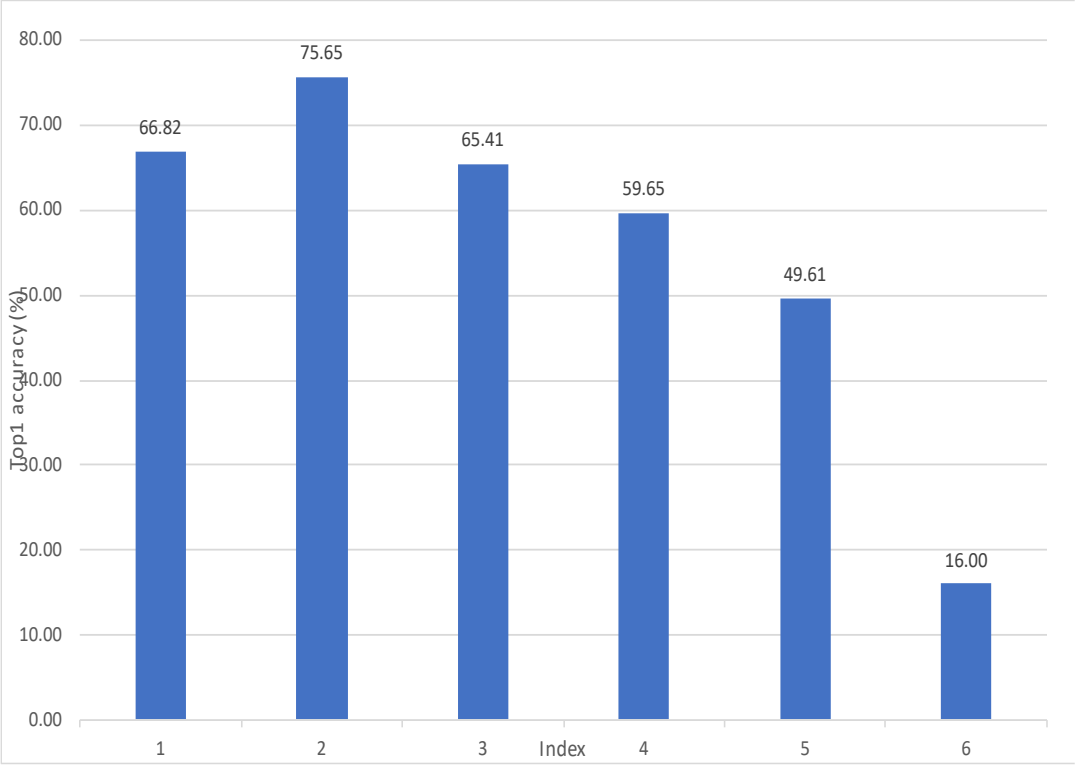

Figure 30 Results: Contribution of real

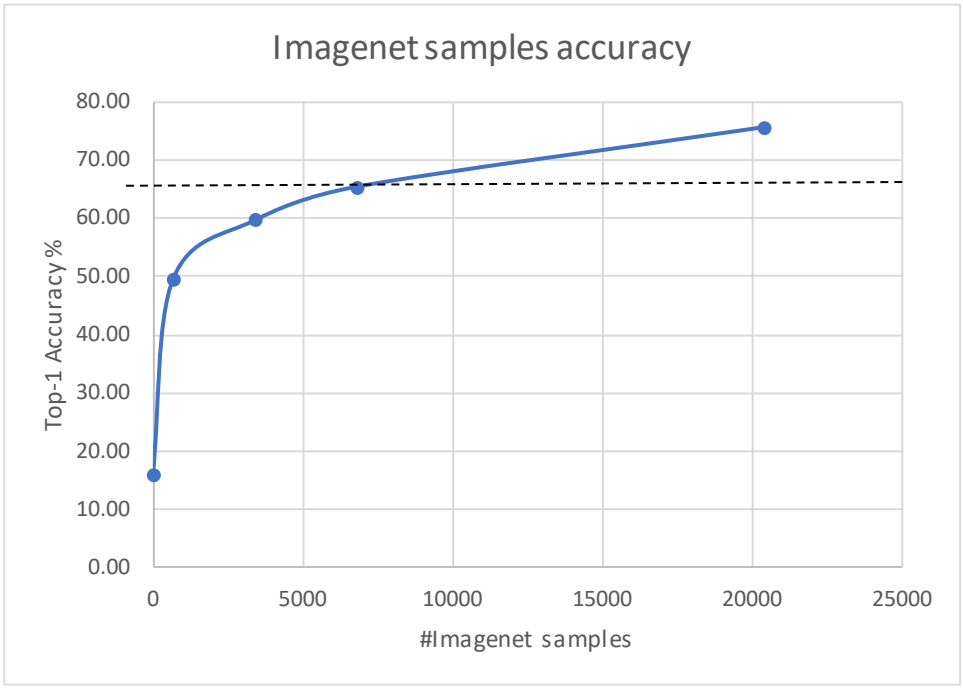

Figure 31 Summary: Contribution of real data. Dashed line represents the baseline

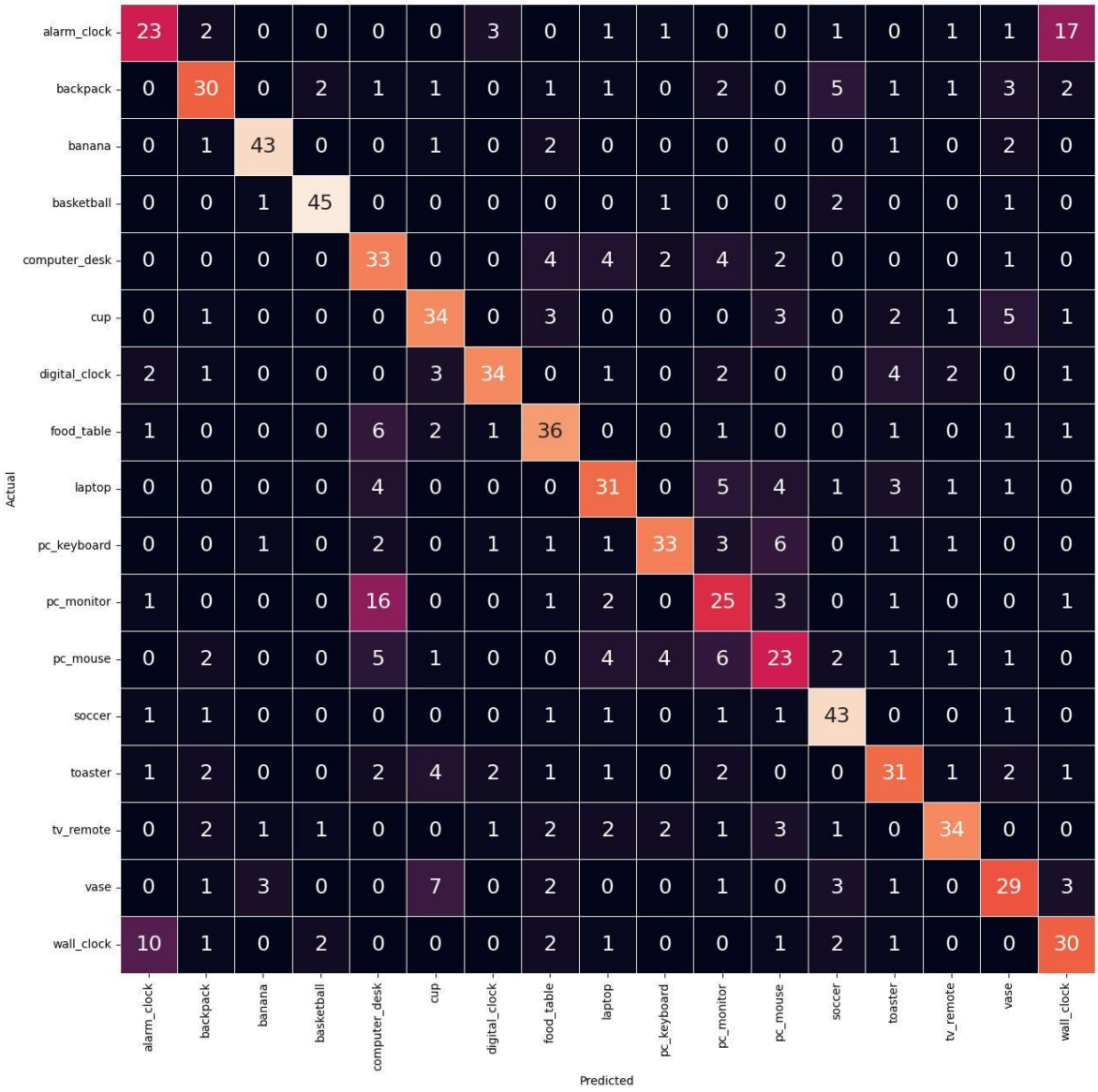

Figure 32 per class confusion matrix mixed: synthetic(100%)+real(33%)

According to Figure 31 we can observe how the real dataset in various sizes together with the full synthetic dataset impacts the performance. It is interesting to note that impact on the performance gain is much more significant in the beginning, with small addition magnitudes, and much less significant with big additions. This observation is also true for synthetic contribution experiments shown in Table 19. Adding real data to the train set shows the network samples from the target domain (test set), practically injecting target domain knowledge into the model. Hence this process is sort of a domain adaptation. Those results show how it is crucial for the network

to learn at least some target domain samples, and even small portions of this knowledge along with the synthetic data lead to fine results. Here, again it is evident that the domain gap is a major contributor of degraded results when trained solely on synthetic data. The dashed line around y=66 represents the performance of the baseline. Therefore starting from this point x=6,800, each real data addition over performs the baseline.

Index	Train	Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
1	Synth hybrid Imagenet Train 1	Imagenet 1	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	70.50	88.79	93.76	70.79	70.50
2	Synth hybrid Imagenet Train 1	Imagenet 2	---	70.50	88.39	94.34	70.91	70.39
3	Synth hybrid Imagenet Train 1	Imagenet 3	---	70.76	88.47	94.53	71.28	70.77
4	Synth hybrid Imagenet Train 1	Imagenet 4	---	70.71	88.99	93.61	71.23	70.70
5	Synth hybrid Imagenet Train 1	Imagenet 5	---	69.48	88.43	94.75	69.75	69.45
6	Synth hybrid Imagenet Train 2	Imagenet 1	---	70.63	89.15	94.05	71.18	70.67
7	Synth hybrid Imagenet Train 2	Imagenet 2	---	69.94	88.34	94.01	70.44	69.96
8	Synth hybrid Imagenet Train 2	Imagenet 3	---	69.93	89.16	94.56	70.28	69.96
9	Synth hybrid Imagenet Train 2	Imagenet 4	---	69.46	87.60	93.41	70.04	69.48
10	Synth hybrid Imagenet Train 2	Imagenet 5	---	70.46	88.48	93.57	70.78	70.49
11	Synth hybrid Imagenet Train 3	Imagenet 1	---	70.38	88.86	94.40	70.77	70.34
12	Synth hybrid Imagenet Train 3	Imagenet 2	---	70.45	89.11	93.90	70.76	70.41
13	Synth hybrid Imagenet Train 3	Imagenet 3	---	68.84	87.86	93.54	69.15	68.80
14	Synth hybrid Imagenet Train 3	Imagenet 4	---	69.31	87.99	93.63	69.69	69.27
15	Synth hybrid Imagenet Train 3	Imagenet 5	---	69.99	88.00	93.59	69.96	69.93
	Standard Deviation			**0.57**	**0.48**	**0.43**	**0.61**	**0.57**

Table 24 Results: results stability and reliability

The results show statistically stable numbers across all the measurements and metrics, proving the results are not just random for this particular experiment setup.

Index	Train	Test	Hyper parameters	accuracy-top1	accuracy-top3	accuracy-top5	precision-top1	recall-top1
	Real #(%)							
1	6,800(33%)	Imagenet	Resnet 50 Optimizer=SGD LR=0.01 LRadjust=True	46.00	67.65	79.88	47.28	46.00
2	3,400(16%)	Imagenet	---	30.82	53.06	66.82	28.94	30.82
3	680(3.3%)	Imagenet	---	21.41	40.24	53.53	17.88	21.41

Table 25 Results: trained on ImageNet only

The results in Table 25 show the model performance when trained and validated on the same portions of ImageNet only as were used for experiments shown in Table 23. We demonstrate those experiments in order to isolate the contribution of the synthetic data achieved in experiments in Table 23. For instance, we can see that the 3.3% of ImageNet (index 3 in Table 25 and the corresponding index 5 in Table 23) boosts the model performance by 49.61%-21.41%=28.2% for top-1 accuracy.

7. Discussion

7.1. Conclusions

The results in this work have shown the model performance significantly degrades for both of the cross domain directions: when model is trained on real and tested on synthetic (see Table 11) and when trained on synthetic and tested on real (see Table 14). Meanwhile, for the same domain directions, specifically: trained and tested on synthetic, the performance results were very good Table 12. Therefore the model overfits to the synthetic data domain and struggles to generalize to the new (real) domain. Unlike the machine, we humans have no problem with performing such transitions – to recognize objects in images that were synthetically created (perform real to synth inference), and supposedly when learning unknown in the real world object representation solely from synthetic and recognizing them in the real life (perform synth to real inference). And it leads to the conclusion that the current deep learning methods are limited in a cross domain generalization. Based on related works, like [23] [16] [24] [22], domain randomization is expected to minimize the domain gap. However, we found out that using only the specific domain randomization techniques used here, did not help to achieve the desired results in case of the model trained solely on synthetic data. Results in Table 23 (experiment index=5) demonstrate that even a neglectable addition of real ImageNet data of total 680 images (3.3% of total ImageNet) to the synthetic training dataset of size 187,928, which leads to the dataset expansion of 0.36% only, improved the model performance from 16% to 49.61% top 1 accuracy. This suggests that incorporating target domain knowledge into the model is a significant step one in any similar situation. Although the domain gap is the most dominant contributor, there are other additional factors that contribute to performance degradation (see 7.3).

The most significant conclusion from this work is that as we have shown it is possible to replace portions of real data with synthetic data and achieve the same results in context of a deep learning

model training for the object classification task. Results in Table 23 (experiment index=3) the case of a full synthetic data set mixed with a real data ImageNet 6,800 (out of 20,400 full ImageNet) samples with the same model implementation and same hyper parameters, produce top 1 accuracy of 65.41% which is almost equal to the baseline with 66.82%. Top 1 accuracy is just an example, but the rest of measured metrics in this experiment are also very close to the baseline. It means that 187,982 training synthetic samples worth of 20,400-6,800=13,600 (66.66%) training real data samples. So for each missing real data sample it is required to compliment it with 187,982/13,600=13.82 synthetic samples. Synthetic vs real data ratio might seem quite big, but synthetic data is generally much cheaper to produce. There are other works that share similar observations and conclusions [19] [20] [10] [21], where synthetic data is complementary to real data.

Experiments performed on the synthetic training set demonstrate that a model trained with images that include backgrounds for object images, produces much better performance results than the ones without a background. It is true both when testing on synthetic data cross category see Table 13, when testing on a real data, where synthetic with background yields top 1 accuracy=16.0 while without=8.82 see Table 14. Background addition contributes to 2 main factors, which might explain these results. First, since most of the general real data from the wild, and in particular the objects photos from the (ImageNet) test set were taken in some context – with a natural background, adding backgrounds to the synthetic data makes it more similar to the target data, therefore this process potentially minimizes the domain gap. Second, because of the online – during the training phase, object image with a background merge algorithm, each object image will get multiple backgrounds within the same training session with a very high probability, leading to a high degree of domain randomization.

From the results obtained during the experiments, it appears that adding more data to the train set improves performance, but the effect is less and less significant based on the size of the additional data. See Figure 25, Figure 31.

7.2. Contribution Estimation

There are lots of benefits of synthetic data over real world data as a machine learning training data source. More than that, synthetic data might potentially boost machine learning research and production development Gartner predicts synthetic data will reach the plateau of productivity in two to five years as one of the technologies in the "innovation trigger" phase of the hype cycle [30]. They also predict that 60% of the data used for the development of AI and analytics projects will be synthetically generated by 2024 [31]. However, as for now, it is a barely explored area of research. Thus, every effort contributes to a deeper understanding of the capabilities and possibilities.

Our work shows that synthetic data can be used to complement the real-world data in computer vision models training with deep learning while achieving satisfying results.

We also propose a unique synthetic data generation method designed for the object classification task. Using this method reduces the amount of computation needed to a minimum, making it a very efficient method. The technique can easily be extended to both object detection and segmentation tasks with minor modifications during the merging with a background phase and auto-labeling.

7.3. Limitations

One of the most obvious problems of the object image with background merge algorithm is that it injects the object in a random way into the background scene. This is unnatural to us humans. A toaster with a kitchen background will most likely be floating in the air rather than being placed on some surface as one would expect from an algorithm Figure 8. It is unclear how and if at all this affects the final performance, since the neural network does not necessarily see the world we do, and therefore "care" about the semantic placements. But in order to overcome this, a much more complicated algorithm that builds a complete 3D environment, taking into account physics, with a semantically valid placement should have been done. Such an approach might be a topic for a dedicated work by itself and it was outside the scope.

Since deep learning algorithms are hungry for diversity, especially when tested on highly diverse in the wild data. Based on 4.4.2, we speculate that the lack of intraclass variance in the synthetic dataset could have a substantial impact on the performance of the model.

Per class instances, the imbalance among different classes in the synthetic dataset is far from ideal. The reason for the imbalance is this is the data we had. This would require a dramatic reduction in number of samples: the smallest class is 28 instances, so it would leave 17*28*26=12,376 samples (6.58% of what we used). But in this work we used the stochastic gradient descent optimization method during all the training sessions, so that the data was randomly sampled, regardless of each class size in the training set, reducing bias towards more populated classes.

Lack of lighting variance. And also the background lighting does not necessarily correspond to the object image lighting, since those are 2 different pictures. It would have been possible for both problems to be resolved if the object images were rendered against a background with a variety of lighting scenarios, in particular an HDRI image background. HDRI emits light, affecting the object in the scene.

The intrinsic parameters of the camera that determine the image objects (lens types, FOV, etc.) are constant in this work, unlike the real world with huge degrees of variance.

7.4. Future Work

It would be interesting to analyze how addressing at least some of the previously mentioned limitations, would contribute. So future work in this aspect would be adding more highly diverse instances for those classes in order to achieve both cross classes balance and a broader intra class variance. Try to render a complete scene – both the objects and the HDRI backgrounds with high variance in lighting and camera parameters.

In this work we introduce synthetic and real data mixing during the same training phase. However there's another possible approach, to pre train on a real and fine tuning on synthetic or vice versa. Also there are different training strategies, like a curriculum strategy introduced in [21] that might be experimented with.

In this work we have chosen to follow the domain randomization strategy in order to reduce the domain gap. In addition to examining the impact of those randomization techniques used, it would be interesting to also explore other randomization techniques. Alternatively, the more complicated to implement domain adaptation approach could be used. For instance transferring each synthetic object image to the real world representation and other methods like introduced in [25] [26] [27].

And finally, this workflow might be extended to other computer vision tasks, like object detection and object segmentation. It will, however, require working with different real datasets like [32] [33] for the baseline, potentially leading to a different classes selection and consequently generating additional synthetic data. The bounding box and mask labels can be derived during the merging phase if following the scheme presented here.

References

[1] Ren, S., He, K., Girshick, R. and Sun, J., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *Advances in neural information processing systems*, pp. 91-99, 2015.

[2] He, K., Gkioxari, G., Dollár, P. and Girshick, R, "Mask R-CNN," *Proceedings of the IEEE international conference on computer vision*, pp. 2961-2969, 2017.

[3] Roh, Y., Heo, G. and Whang, S.E., "A survey on data collection for machine learning: a big data-ai integration perspective," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[4] Heller, N., Dean, J. and Papanikolopoulos, N., "Imperfect Segmentation Labels: How Much Do They Matter?," *Intravascular Imaging and Computer Assisted Stenting and Large-Scale Annotation of Biomedical Data and Expert Label Synthesis*, pp. 112-120, 2018.

[5] Blender. [Online]. https://www.blender.org/

[6] Unity. [Online]. https://unity.com/

[7] Unreal. [Online]. https://www.unrealengine.com/

[8] Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D. and Brox, T., "FlowNet: Learning optical flow with convolutional networks," *"Flownet: Learning optical flow with convolutional networks." In Proceedings of the IEEE international conference on computer vision*, pp. 2758-2766, 2015.

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

[10] Cabon, Y., Murray, N. and Humenberger, M., "VIRTUAL KITTI 2," *arXiv preprint arXiv:2001.10773*, 2020.

[11] Qiu, W. and Yuille, A., "UnrealCV: Connecting computer vision to Unreal Engine," *European Conference on Computer Vision*, pp. 909-916, 2016.

[12] Ros, G., Sellart, L., Materzynska, J., Vazquez, D. and Lopez, A.M., "The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3234-3243, 2016.

[13] Richter, S.R., Vineet, V., Roth, S. and Koltun, V., "Playing for data: Ground truth from computer games," *European conference on computer vision*, pp. 102-118, 2016.

[14] Müller, M., Casser, V., Lahoud, J., Smith, N. and Ghanem, B., "Sim4CV: A photo-realistic simulator for computer vision applications," *International Journal of Computer Vision 126, no. 9 (2018)*, pp. 902-919, 2017.

[15] Jalal, M., Spjut, J., Boudaoud, B. and Betke, M., "SIDOD: A Synthetic Image Dataset for 3D Object Pose Recognition with Distractors," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0-0, 2019.

[16] Tremblay, J., To, T. and Birchfield, S., "Falling Things: A Synthetic Dataset for 3D Object Detection and Pose Estimation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, pp. 2038-2041, 2018.

[17] Shah, S., Dey, D., Lovett, C. and Kapoor, A., "AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles," *Field and service robotics*, pp. 621-635, 2017.

[18] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A. and Koltun, V., "CARLA: An Open Urban Driving Simulator," *arXiv preprint arXiv:1711.03938*, 2017.

[19] Beery, S., Liu, Y., Morris, D., Piavis, J., Kapoor, A., Joshi, N., Meister, M. and Perona, P., "Synthetic Examples Improve Generalization for Rare Classes," *The IEEE Winter Conference on Applications of Computer Vision*, pp. 863-873, 2019.

[20] Hattori, H., Naresh Boddeti, V., Kitani, K.M. and Kanade, T., "Learning Scene-Specific Pedestrian Detectors without Real Data," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3819-3827, 2015.

[21] Hinterstoisser, S., Pauly, O., Heibel, H., Marek, M. and Bokeloh, M., "An Annotation Saved is an Annotation Earned: Using Fully Synthetic Training for Object Instance Detection," *arXiv preprint arXiv:1902.09967*, 2019.

[22] Hodaň, T., Vineet, V., Gal, R., Shalev, E., Hanzelka, J., Connell, T., Urbina, P., Sinha, S.N. and Guenter, B., "Photorealistic image synthesis for object instance detection," *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 66-70, 2019.

[23] Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W. and Abbeel, P., "Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 23-30, 2017.

[24] Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S. and Birchfield, S., "Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain

Randomization," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 969-977, 2018.

[25] Csurka, G., "A comprehensive survey on domain adaptation for visual applications," *Domain adaptation in computer vision applications*, pp. 1-35, 2017.

[26] Peng, X., Usman, B., Saito, K., Kaushik, N., Hoffman, J. and Saenko, K, "Syn2Real: A New Benchmark for Synthetic-to-Real Visual Domain Adaptation," *arXiv preprint arXiv:1806.09755*, 2018.

[27] Peng, X., Usman, B., Kaushik, N., Wang, D., Hoffman, J. and Saenko, K., "VisDA: A Synthetic-to-Real Benchmark for Visual Domain Adaptation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 2021-2026, 2018.

[28] Dong W, Socher R, Li LJ, Li K, Fei-Fei L. Deng J, "Imagenet: A large-scale hierarchical image database," *IEEE conference on computer vision and pattern recognition*, pp. 248-255, 2009.

[29] A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba B. Zhou, "Places: A 10 million Image Database for Scene Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[30] (2021) Gartner. [Online]. https://www.gartner.com/en/articles/the-4-trends-that-prevail-on-the-gartner-hype-cycle-for-ai-2021

[31] Gartner blog. [Online]. https://blogs.gartner.com/andrew_white/2021/07/24/by-2024-60-of-the-data-used-for-the-development-of-ai-and-analytics-projects-will-be-synthetically-generated/

[32] Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Kolesnikov, A. and Duerig, T., "The Open Images Dataset V4," *International Journal of Computer Vision (2020)*, pp. 1-26, 2020.

[33] Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L., "Microsoft coco: Common objects in context," *In European conference on computer vision*, pp. 740-755, 2014.

[34] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. and Desmaison, A., "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, pp. 8026-8037, 2019.

[35] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., "Tensorflow: A system for large-scale machine learning," *12th*

[36] K., Engstrom, L., Ilyas, A. and Madry, A. Xiao, "Noise or signal: The role of image backgrounds in object recognition," *arXiv preprint arXiv*, 2020.

[37] V., Shin, H. S., Tsourdos, A., & Colosimo, N. Varatharasan, "Improving Learning Effectiveness For Object Detection and Classification in Cluttered Backgrounds," *2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS)*, no. IEEE, pp. 78-85, 2019.

[38] (2021) Gartner. [Online]. https://www.gartner.com/en/articles/the-4-trends-that-prevail-on-the-gartner-hype-cycle-for-ai-2021