Minimizing Gaps Between Synthetic And Real Datasets

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Minimizing Gaps Between Synthetic And Real Datasets

Master Thesis

Mengtao Zhang

September 08, 2023

Advisors: Prof. Dr. Vechev, Dr. Bielik, Mr. Dhall

Department of Computer Science, ETH Zürich

Abstract

Creating synthetic images that are of high quality is a crucial step for many deep learning projects, especially when real data are either limited or too expensive to acquire. Despite its importance, there has been a noticeable gap in the exploration of efficient techniques in the synthetic image generation field, except for generating photo-realistic samples of high cost. In this study, we bridge this gap by investigating the key elements and features that should be maintained and aligned in synthetic datasets to mirror the properties of real datasets. From our findings, we have developed practical guidelines that simplify the process of creating synthetic datasets that can stand up to their real counterparts, particularly in the realm of object detection tasks. We create a reusable framework that can guide future research and developments in this area.

Contents

Chapter 1

Introduction

Deep learning has significantly changed computer vision, leading to improvements in image classification, object detection, and semantic segmentation [1]. However, there's still a big challenge—the ongoing need for a lot of high-quality labeled data, especially for specialized areas like medical imaging [1]. Machine learning and deep learning are highly dependent on having a lot of labeled data and strong computing power. While there are methods using unsupervised learning with unlabeled data, these models often aren't as effective as the supervised ones. This difference is especially clear in areas like medical diagnostics [2].

Acquiring those large-scale, high-quality labeled datasets for computer vision tasks can be a challenging and expensive endeavor [3]. Many real-world applications, such as hand gesture labeling for sign language recognition, involve labor-intensive efforts and expert annotators, making the data collection process resource-intensive and time-consuming. Moreover, data in practical scenarios often originates from multiple sources, leading to domain shifts between the training and deployment environments. For instance, in a task like car damage detection for insurance claims, data might come from various sources, including smartphone images, surveillance cameras, and professional photographs, resulting in variations in image quality, lighting conditions, and perspectives.

One promising solution to tackle these challenges is the use of synthetic data. Synthetic data can be generated using computer graphics or simulation techniques, offering a cost-effective alternative to address data scarcity and diversify datasets. For example in hospitals, synthetic data can be utilized to augment the limited real-world dataset, creating a comprehensive set of medical images with different patient conditions, as the example shown in figure 1.1. This not only enhances the dataset's diversity but also reduces the need for manual labeling, as synthetic data can come with predefined attributes and annotations [5]. In addition to that, using synthetic data also

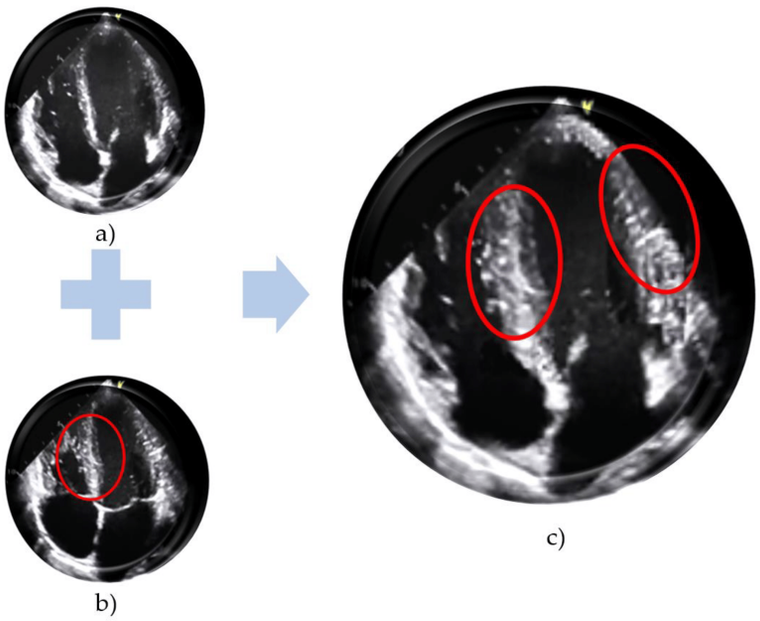


Figure 1.1: An example of synthetic medical images. From the image of a healthy heart when its mitral valves open (a) and closed (b), a synthetic heart image with the open mitral valve and thickened LV (c) is made.[4]

offers additional advantages, such as the ability to control the appearance, lighting, and background of objects or scenes when we use handcrafting or engine-based rendering pipelines, described in more detail in the related works chapter 2.1. This level of control is especially valuable when simulating complex or rare scenarios that are challenging to capture in real-world datasets. Furthermore, synthetic data can provide ground truth labels for free, allowing for large-scale data generation and model training without the laborious annotation process [6]. Typical examples are camera-like image data. These images range from digital alterations to ones from virtual settings [7], and from combined images [8] to very realistic computer-made visuals [9].

1.1 Synthetic Datasets: Benefits and Limitations

The benefit of synthetic data is its ability to speed up the data-making and labeling process, often being faster and cheaper than manual methods. However, it's important to note that not all synthetic methods automatically label data; some need a lot of human help [10]. For instance, a GAN [10] may be able to produce realistic images of certain objects of interest, but it is hard for them to produce the correct bounding box annotations alongside when you perform an object detection task. Compared to traditional data

collection, synthetic methods are usually quicker and more cost-effective. Challenges in gathering real data include privacy issues, rules, and practical hurdles. Manual labeling can be time-consuming, and human mistakes can reduce data quality, affecting model training [11]. Interestingly, starting an automated labeling system requires a lot of labeled data, leading to a kind of loop. But synthetic data gives better control over data quality, allowing for detailed labels and insights that might be hard to get in real situations.

The integration of synthetic data in deep computer vision has indeed provided a vital means to address the need for labeled data. However, it is not without its limitations, particularly concerning the disparity between synthetic data and real-world data [12]. One of the primary limitations lies in the realism of synthetic images. Although synthetic images can be crafted to resemble real-world counterparts, they often exhibit subtle discrepancies in lighting, texture, and geometry, leading to a lack of perfect alignment with real-world images. While it is possible to control those parameters explicitly/implicitly, it often remains unclear what the essential attributes and characteristics in those real data that need to be captured and preserved when generating synthetic image samples and what are not essential and could vary under different contexts. As a result, models trained on synthetic data may excel on synthetic images but struggle to generalize effectively to real-world scenarios [7]. Figuring out the most important attributes and giving a clear guideline to produce cheap but effective synthetic image samples is the main focus and contribution of this thesis.

Another significant limitation arises from the restricted diversity of synthetic images. The diversity of synthetic data generation depends on the virtual environments and objects utilized in the process. Despite being able to generate a large volume of synthetic images, they may not encompass the full spectrum of variations present in real-world data. Consequently, models trained solely on synthetic data may falter when confronted with unseen variations or rare occurrences.

When considering the deployment of synthetic data in practical scenarios, several factors must be accounted for. One of the foundational considerations is the environment where the trained model will be deployed. For instance, a model trained on synthetic data for traffic monitoring needs to be validated against real-world traffic scenarios to ensure its robustness and accuracy.

Another lesser-discussed issue is the computing power needed to make synthetic data. Making big synthetic datasets in a reasonable time requires strong computing resources, which might not be available to everyone. While many studies praise synthetic data, only a few discuss the computing needs and time required to make such datasets. This point is crucial when considering the use of synthetic data in different areas.

Thanks to many studies, computer vision has seen benefits by using synthetic

images to enhance real datasets. Efforts are also being made to lessen the gap between training and testing datasets with model-centric domain generalization methods as well as data-centric methods, see examples in the related works section 2.2.1.

Moreover, given the rarity of real-world data, what little there is can be reserved primarily for validation purposes. Models trained on synthetic data can be tested iteratively against this small but invaluable set, ensuring that learnings from each test phase are looped back to refine the training process [13]. This iterative feedback can be pivotal in bridging the synthetic and real-world data gap.

1.2 Our Contribution: A Novel Data-Centric Approach

As our exploration in the background section revealed, the demand for high-quality labeled data remains a primary challenge, especially in specialized domains [1]. Synthetic data emerges as a promising solution, offering controlled data generation, a cost-effective alternative, and often, predefined annotations [5, 6]. Despite these advantages, the gap between synthetic and real-world data is palpable, with evident discrepancies in realism, diversity, and intrinsic noise representation [12, 7]. Such disparities necessitate approaches that can bring synthetic data more in line with real-world data.

To this end, we introduce a data-centric methodology to address the significant challenge of reconciling the differences between synthetic and real-world data in computer vision. The underpinning of our work is rooted in dataset engineering techniques. Unlike many methods that often prioritize model architectures or training techniques, our methodology delves deeply into the essence of data itself, recognizing the foundational role it plays in the overall performance and generalization of machine learning models. The following encapsulates our contributions:

1. **Novel Guidelines for Generating Synthetic Images:** Rooted in real dataset insights, our proposed approach optimizes synthetic data generation from two distinct angles: metadata and feature perspectives. In particular, we align the distributions of metadata and feature representations between the synthetic and real datasets, as an alternative approach to directly align the data distribution of the synthetic and real image samples, which is the idea behind generating synthetic images that are as photo-realistic as possible. Through rigorous benchmarking, we have established a set of guidelines aimed at refining future synthetic image generation endeavors.
2. **Enhanced Performance with Synthetic Datasets:** We have created synthetic datasets that, through meticulous refinement, exhibit performance akin to models trained on real-world data across diverse

computer vision algorithms. In particular, object detection algorithms trained solely on generated synthetic datasets outperform the same algorithms trained on real datasets 50% of the time and achieved over 1000% improvement in MAP score [14] compared to the baseline synthetic datasets. This achievement underscores our methodology's effectiveness in bridging the synthetic-real data performance chasm.

3. **Efficient Synthetic Image Generation Pipeline:** We introduce a semi-automated pipeline tailored for synthetic image generation. This framework semi-autonomously selects optimal sampling strategies based on the inputs, consequently bridging the domain gap and bolstering the generalizability of computer vision models. This pipeline can generate synthetic datasets which possess a very similar distribution in selected dataset attributes, e.g. the area of the objects of interest in the images, given the desired distribution.

4. **Correlation Analysis for Model Performance:** Delving deep into synthetic datasets, we explore the linkage between distribution distances and model performance discrepancies. We observe negative correlations between model predictions on the real samples in the test dataset with the distances of real samples to the clusters in synthetic datasets across different objects of interest. In particular, the prediction on the test sample tends to be less accurate when the test sample does not have enough similar samples in the training samples. This scrutiny provides vital insights into factors affecting algorithm performance, proving instrumental for informed dataset engineering.

5. **Benchmarking of Feature Space Representations:** Our investigation extends to comparing various feature space representations omitted from both synthetic and real samples. The differences in the quality of feature extractors regarding the information the features contain are significant, especially between pre-trained feature extractors and trained extractors, where the difference could be over 500%. This analysis unveils the intricacies of different feature representations and their implications on domain alignment.

Our approaches to bridging the gap between real datasets and synthetic datasets are novel and effective. These approaches, to our knowledge, have neither been implemented nor thoroughly studied in the existing literature, making our contributions both innovative and of importance for future progression. In particular, we focus on four critical aspects: metadata distribution alignment, elaborated in 4.6, feature space distribution alignment, elaborated in 4.7, cheap and easy-to-use semi-automated metadata alignment pipeline, described in 4.6.3 and heuristic guidelines towards cheap but meaningful synthetic image generation.

Chapter 2

Related Works

We navigate the extensive landscape of relevant literature, dissecting contributions from two core areas that align with our objectives. Firstly, we delve into the intricacies of synthetic image generation pipelines, tracing the evolution of techniques and tools that have emerged over time. Secondly, we focus on domain generalization underscored by data enhancement methodologies, which offer insights into how diverse datasets can be harnessed to achieve improved generalization. In the ensuing section, we articulate our distinctive approach, underscoring the nuances that set our methodologies apart from those prevalent in the existing literature. This dual examination not only sets the stage for our research but also contextualizes our contribution within the broader academic dialogue.

2.1 Methods of synthetic image generation

There are many means to generate synthetic data with different methodologies suited to different tasks and applications. However, most methods to generate synthetic data for computer vision can be grouped into the following categories.

2.1.1 Handcrafted synthetic images

Handcrafting synthetic images represents a foundational method of data synthesis. Whether it's a composite image or a 3D environment, each data instance is produced one by one to compile a complete dataset [15]. This technique is considerably time-intensive and restricts the amount of data that can be crafted. The labeling and annotation of such handcrafted data typically demand further hands-on efforts. This negates some of the main advantages of synthetic data, which are large-scale production and automated feature labeling. Still, handcrafting synthetic images has its place in certain scenarios. Typically, the handcrafting is done by an artist who creates the 3D models

from scratch and composes them in a meaningful way. However, in tasks like object detection, the samples often focus on specific object subsets within that realm. Therefore, handcrafting specific, smaller virtual spaces might be more advantageous than auto-generating expansive ones that might go underutilized.

2.1.2 Engine-based automated rendering pipeline

In recent times, there has been a noticeable shift towards leveraging engine-based rendering pipelines for the generation of synthetic data. Research is increasingly focusing on game engines like Unity and Unreal Engine, alongside 3D modelling tools such as Blender [16]. These platforms, given their advanced capabilities, provide a more efficient alternative to building a 3D virtual environment from the ground up. They offer researchers the convenience of using pre-existing assets, simulating diverse scenarios with virtual cameras, implementing artificial lighting, and employing various other functionalities that ease the process of virtual space creation. Owing to their core design, game engines are adept at curating virtual realities. Furthermore, continuous updates, extensions, and third-party tie-ups have made it possible to directly link these engines with networks for training. This capability allows for the generation of a substantial volume of realistic data, which finds applications in areas like robot simulations, aerial aircraft identification [17], vehicle detection, and identifying inattentive pedestrians [18]. Often, these engines can auto-generate exhaustive ground truth information, such as segmentation masks and depth maps.

Those engines usually have considerably elevated the process of synthetic image sampling, largely due to the advanced scripting tools they encompass. Tools like BlenderProc [19] in the Blender ecosystem and Replicator for game engines have emerged as front-runners in this domain, enabling users to automate intricate synthetic image generation workflows. These tools facilitate comprehensive sampling strategies, ensuring that images are not just synthesized, but also cater to a wide range of scenarios and environmental conditions. For instance, BlenderProc effortlessly combines scene creation, object manipulation, camera and light positioning, and rendering into streamlined scripts, which significantly expedites the synthetic data generation process. On the other hand, Replicator offers versatility by producing varied instances of objects, lighting conditions, and backgrounds, ensuring a rich dataset. Such scripting capabilities underscore the potential of these platforms to support large-scale and diverse synthetic dataset generation, making them invaluable assets in the rapidly evolving landscape of computer vision and machine learning.

2.1.3 Generative Models

Over the past few years, significant strides have been made in the domain of synthetic image generation, primarily due to the advent of deep generative models. These models are designed to yield images that closely mirror the distribution of a designated dataset. Three models that stand out in driving this progress are the Variational Autoencoders (VAEs) [20], Generative Adversarial Networks (GANs) [21], and Diffusion Models [22].

Variational Autoencoders (VAEs) operate by transforming an input image into a compact latent representation, which is subsequently decoded to regenerate an image. A hallmark of VAEs is the continuous and fluid nature of their latent space, permitting the generation of images with nuanced variations. Such a feature is instrumental for tasks that demand intricate image variations. Nevertheless, a recurrent challenge with VAEs is their tendency to yield slightly blurred images, which can be less than ideal for scenarios that necessitate crisp and clear visual outputs.

On the other hand, Generative Adversarial Networks (GANs) rely on a dual neural network setup involving a generator and a discriminator. The generator is responsible for synthesizing images, while the discriminator's role is to differentiate between authentic and synthesized images. As the training evolves, the generator refines its capability to render increasingly realistic images. GANs have gained acclaim for their prowess in generating high-fidelity images that adeptly encapsulate the nuances of the training data. However, they're not devoid of challenges. A notable one is their training sensitivity, potentially leading to complications like mode collapse.

Diffusion Models, which are relatively nascent compared to VAEs and GANs, fabricate images by iteratively refining a preliminary random stimulus, echoing a diffusion procedure. Such a method can be particularly pertinent for tasks where there's value in visualizing the gradual image evolution. But a caveat with Diffusion Models is the extended duration for image generation due to its iterative approach.

While deep generative models serve as potent instruments for synthetic image generation, they also usher in certain challenges. For instance, the computational intensity of training, especially for GANs, often mandates substantial resources. They also predominantly depend on voluminous training datasets, and the uniformity in the quality of the produced images can occasionally waver. Furthermore, ensuring precise control over particular image traits can be intricate, and the resultant synthetic images may be devoid of annotations, posing challenges for applications necessitating labeled datasets.

2.1.4 Synthetic image collections from Games

Recently, realistic 3D video games have been recognized as an innovative avenue for generating synthetic image samples across multiple sectors within artificial intelligence research and deployment. Notable titles in this regard include "Counter-Strike: Global Offensive" (CSGO [23] and "Grand Theft Auto" (GTA) [24]. Both games, celebrated for their sophisticated graphics and intricate world design, present a rich data source yet to be fully exploited. An example of the GTA dataset is shown in figure 2.1.

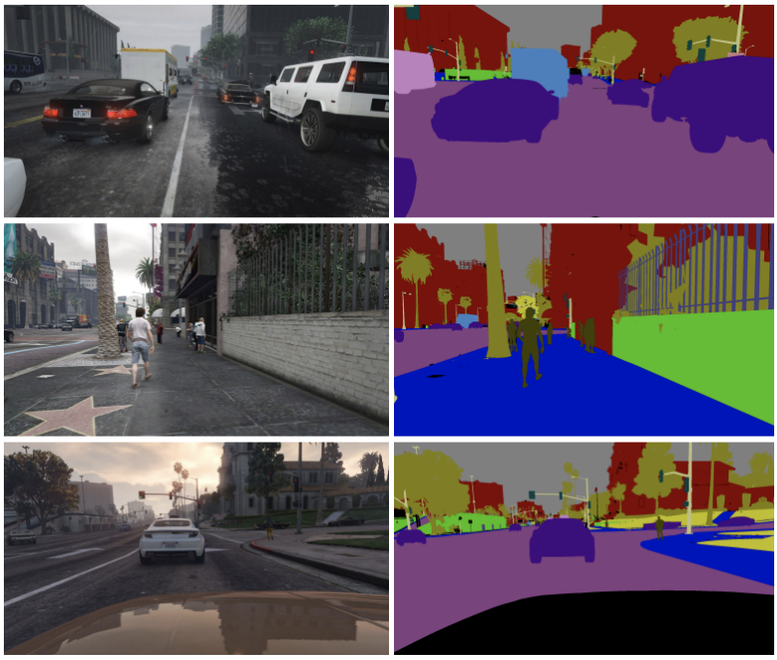


Figure 2.1: GTA images used for segmentation [25]

Consider "GTA," a game marked by its vast open-world design. It exhibits a plethora of scenarios, terrains, and in-game characters, all of which can be harnessed to craft synthetic datasets [25]. Such "GTA datasets" are especially potent for endeavors like simulating autonomous driving or analyzing pedestrian behavior, thanks to the game's bustling urban landscapes, intricate traffic mechanics, and dynamic non-player character interactions. In contrast, "CSGO," distinguished by its comprehensive arsenal of weaponry and gear, has facilitated the development of the "CSGO weapon datasets." These have emerged as assets for research areas emphasizing object detection, especially in contexts where weapon identification is paramount.

Leveraging datasets stemming from these video games bestows multiple benefits. The cost-effectiveness stands out prominently. Procuring image data from these platforms is considerably more economical than orchestrating

real-world datasets, which entail extensive image acquisition, annotation, and refinement processes [25]. The meticulous physics and lighting mechanics integrated into these games, paired with their state-of-the-art graphics, guarantee the resultant images are of impeccable quality. Given the orchestrated nature of these virtual environments, crafting specific situations or edge cases becomes less cumbersome, fostering dataset diversity.

Nevertheless, this approach isn't devoid of hurdles. A salient challenge is annotation. Even though these games render meticulous visuals, they don't innately provide the nuanced labels pivotal for myriad machine learning applications. This often necessitates auxiliary post-processing and hands-on annotation, partially offsetting the cost advantages. However, the horizon seems optimistic. Recent breakthroughs, like the advent of the Segment Anything [26] model by Meta, suggest viable remedies to such annotation dilemmas. Implementing this model empowers researchers to glean finer object and environment labels from synthetic imagery, amplifying the worth of game-sourced datasets.

2.2 Domain Generalization

In the realm of synthetic datasets, a pervasive challenge encountered when adapting to real-world tasks is the discernible distribution disparity between synthetic and actual data. At the heart of addressing this divergence lies domain generalization [27]. This paradigm emphasizes training algorithms across diverse domains, instilling in them the capacity to aptly adapt to and function in unfamiliar territories. The overarching objective is to distil and harness the collective knowledge spanning these domains, repurposing it to navigate uncharted ones. Notably, this is executed without the need to tap into the target domain's data during the training phase, clearly demarcating it from domain adaptation, which often resorts to using unlabelled data from the intended domain.

This ethos underpinning domain generalization echoes the quintessential aspiration of machine learning: devising models adept at generalizing to data they haven't encountered before. Strategies employed to this end encompass meta-learning techniques [28], where models are primed to swiftly acclimatize to novel tasks, to architectural refinements that facilitate the extraction of domain-agnostic features [29].

Of late, the applicability of domain generalization has become increasingly salient in reconciling differences between synthetic and real-world data. The challenge of synthetic-to-real generalization materializes when training on simulated or computer-generated data, a preference due to its plentiful availability and annotation convenience, does not adequately capture real-world intricacies. In this context, domain generalization offers a sturdy scaffold,

ensuring models honed on synthetic datasets retain their effectiveness upon real-world deployment. Such congruence is indispensable in domains like robotics and medical imaging or autonomous driving [30], where obtaining real-world data may be limited, costly, or fraught with risk. In such situations, domain generalization emerges as a pivotal anchor, solidifying the tangible benefits of training on synthetic data without undermining a model's real-world proficiency. Broadly, the techniques under domain generalization seeking to bridge the synthetic-real chasm can be stratified into two main streams: model-centric strategies and dataset-centric methodologies.

2.2.1 Model-centric approaches

Model-centric approaches center on the model itself, aiming to enhance its internal structure, functionality, and robustness. These methods emphasize architectures, losses, and regularization techniques [31] that intrinsically improve the model's ability to understand underlying patterns across varying domains, thereby reducing the domain-specific biases. The model-centric paradigm advocates that if a model can be designed to inherently capture universal patterns irrespective of the data domain, then it can robustly generalize to new, unseen domains. By focusing on the model's design and training process, researchers aim to ensure that the learned representations are more invariant and less sensitive to domain-specific noise. Such approaches may employ techniques ranging from architectural modifications, like adaptive layers or components, to regularization methods that encourage domain-invariance in the learned features. In the context of synthetic-to-real generalization, where models are trained on synthetic data and deployed on real-world scenarios, it becomes possible to attenuate the discrepancies between synthetic training data and real-world application scenarios by emphasizing model structures and training methodologies that inherently understand and prioritize domain-agnostic features.

Zdravko Marinov and Alina Roitberg proposed a novel approach to minimizing gaps between synthetic and real domains using automatic modality selection [32]. The authors propose an unsupervised modality selection method, ModSelect, which emphasizes the importance of the right modality when designing multimodal systems. The intent is to determine which modalities contribute positively to cross-domain activity recognition and automatically select them without the need for ground-truth labels. This unsupervised modality selection approach is built on the correlation between predictions of different unimodal classifiers and the domain discrepancy of their embeddings. It computes modality selection thresholds to choose only those modalities that have a high correlation and a low domain discrepancy. This highlights the method's capability to distinguish between useful and non-useful sources of information, especially when dealing with a significant

distributional shift like the synthetic-to-real gap.

Wuyang Chen and Zhiding Yu [33] proposed an automated synthetic-to-real generalization by doing knowledge distillation from ImageNet to synthetic datasets and Reinforcement-learning-based optimization procedures. The paper introduces a fresh angle to address synthetic-to-real generalization by conceptualizing it as a lifelong learning challenge. By emphasizing the representation similarities between models trained on synthetic data and the ImageNet pre-trained model, the research posits this similarity as a gauge for generalization capabilities. This is visualized in the figure 2.2. In addition, the paper also proposes a novel automated hyperparameter framework, a Reinforcement Learning-based learning-to-optimize strategy. This technique aims to supersede the convoluted manual layer-wise learning rate adjustments with an automated system. Comparative experiments vouch for RL-L2O's superiority over hand-crafted decisions, highlighting its capacity to derive comprehensible learning rate strategies.

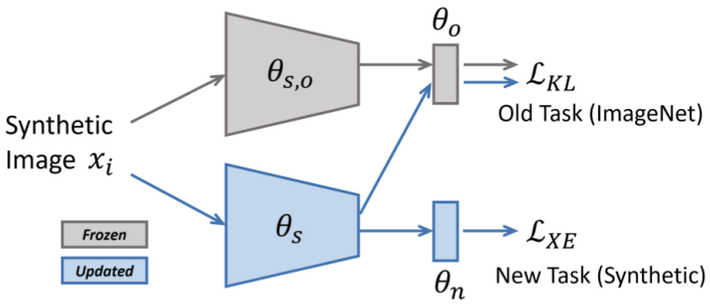


Figure 2.2: Knowledge Distillation in synthetic-to-real generalization

2.2.2 Data-centric approach

While the model-centric approach offers profound insights into tailoring model architectures and training strategies for domain generalization, shifting our gaze to the data itself unveils another avenue of exploration: the data-centric approach. Focusing on the very essence of what our models consume, data-centric strategies emphasize the modification, enhancement, or expansion of training data to better equip models for unseen domains. Techniques such as data augmentation, re-sampling, and domain randomization become the key players [34]. For instance, by introducing variations in lighting, orientation, or background noise, data augmentation can make the model invariant to such changes. Similarly, domain randomization might scatter a spectrum of domain-specific features into training data, encouraging the model to focus on the most consistent and generalizable patterns

[35]. In this section, we'll delve deeper into the related works regarding how data enhancement techniques provide a robust foundation for domain generalization, making models more adaptable to unfamiliar territories.

The most fundamental one among those data-centric approaches would be using proper image preprocessing techniques. An example of using such techniques is shown in figure 2.3.

For instance, Yichun Shi and Xiang Yu [36] proposed a domain-invariant feature learning framework by doing carefully chosen data enhancements on the training data using data synthesizing techniques. In particular, the paper introduces techniques to make models better at recognizing images, especially in unpredictable situations. They address challenges like when faces are turned slightly away, are blurry, or partly hidden. To simulate blurry photos, they used a technique that's like smearing the image a bit. To represent situations where parts of an image might be blocked or hidden, they divided pictures into small sections and darkened some of them. Lastly, for pictures where faces aren't looking directly at the camera, they used a tool called PRNet to adjust the angle of the face. To keep things random and more lifelike, they mixed these changes, applying them to images with a one-in-three chance.

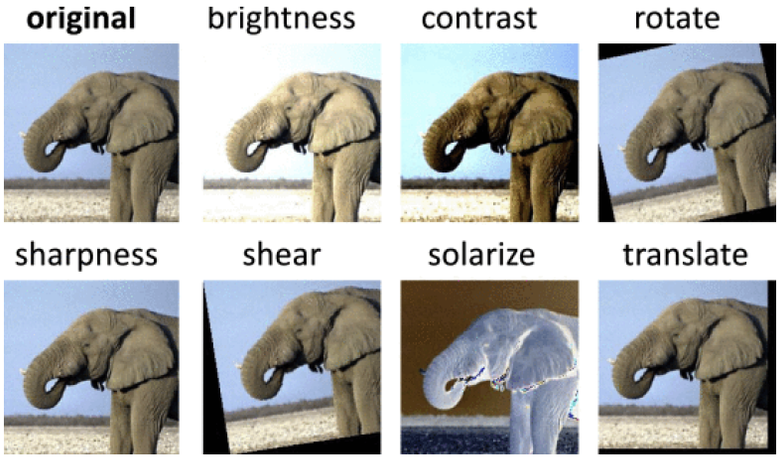


Figure 2.3: Variation in image-preprocessing [27]

Another widely used approach is the adversarial data augmentation with gradient information, inspired by the Goodfellow [37] because gradient perturbation gives the most difficult perturbation for any deep learning model. The paper from Riccardo Volpi and Hongseok Namkoong [38] introduced an approach that uses adversarial data augmentation in domain generalization. In simpler terms, they generate challenging, synthetic training samples that push the model to learn more robust features. By doing so, the

model becomes better equipped to handle diverse and previously unseen data. The methodology behind this involves using a min-max optimization strategy. An adversarial network generates difficult examples appending gradient information to the samples, while the main network tries to correctly classify these challenging samples. This continuous back-and-forth between the two networks ensures the primary model becomes more versatile and adaptive.

Other people try to do adversarial data augmentation with domain-specific information, which is called domain-guided perturbation. Shiv Shankar and Vihari Piratla introduced a Bayesian setting to this matter and viewed adversarial data augmentation as a Bayesian sampling over the domain [39]. Instead of adding adversarial noises to samples solely using gradient information from the task-specific classifier, they also used gradient information from the domain-specific classifier. To generate new training instances that mimic data from a different domain, the idea is to slightly modify or ”perturb” these causative domain-related features. Training this domain classifier, the authors capture the gradient (rate of change) of the loss concerning the input data. This gradient essentially points in the direction of the greatest domain change. By perturbing the data in the direction of this gradient, they can create new instances that look like they come from a different domain. This novel method allows the direct perturbation of inputs without having to separate or remix domain signals or make various assumptions about distributions.

Chapter 3

Overview

The universal approximation theorem states that a neural network can potentially approximate any function within its training dataset [40]. While this can be an advantage, it also makes the network vulnerable to unfamiliar data distributions, especially when using synthetic data in computer vision applications.

In this research, we shift our focus towards a data-centric approach, emphasizing the importance of dataset engineering over simply tweaking model architectures. Our goal is to develop strategies that bridge the gap between synthetic and real datasets, enhancing the performance and reliability of computer vision algorithms in real-world scenarios.

3.1 Overview

Central to our proposal is a focus on advancing dataset engineering techniques. Our objective is to enhance synthetic datasets so that they mirror the complexities encountered in real-world scenarios, thereby narrowing the divergence between these two domains. The ambition here is dual-faceted: firstly, to elevate the authenticity of synthetic data in line with real-world data, and secondly, to boost the adaptability and resilience of computer vision algorithms. An overview of our approach is shown in figure 3.1. We will explain each component in this figure further in the following sections and chapters.

As the overview figure 3.1 shown, we propose a four-fold method to improve dataset quality and alignment between generated synthetic and given real datasets: using heuristic Metadata Distribution Alignment to better understand and organize data, implementing Feature Space Distribution Alignment to create a smoother transition between synthetic and real data feature spaces, and employing Correlation Analysis to study the relation-

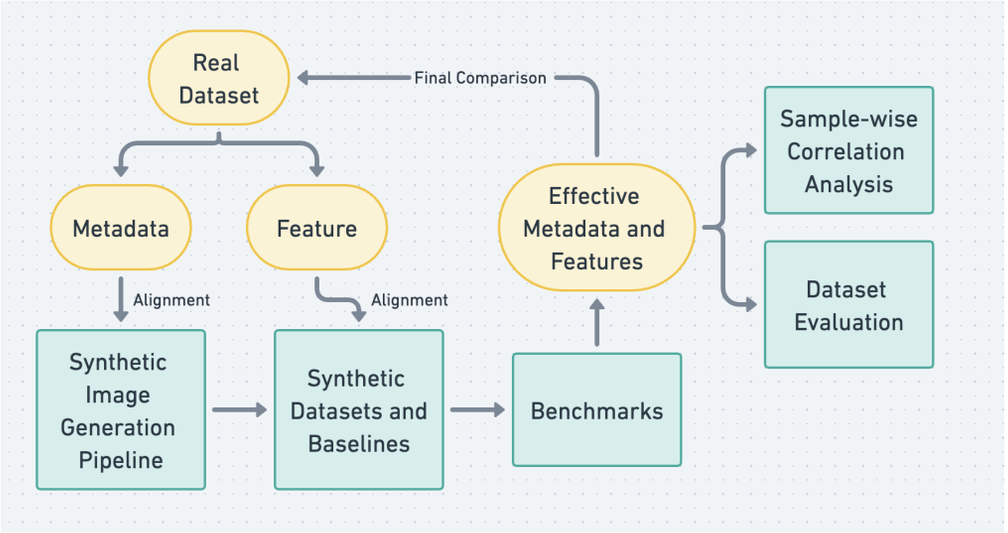


Figure 3.1: A high-level overview of our approaches. We begin with aligning metadata and feature distributions in the synthetic generation pipeline(an explanation of alignments can be found in 3.1, 4.7 and 4.6), verify the effectiveness with benchmarking and compare the final results with the real dataset. The outcomes of benchmarks would then be used for correlation analysis and dataset quality evaluation.

ships between different data attributes. For exploring the distributions, we use different approaches. For one-dimensional metadata distributions, we view the distribution as histograms and align the histograms between synthetic and real datasets. For high-dimensional feature distributions, we fit the distribution with the non-parametric model and align the distributions by enclosing the distances between the fitted non-parametric models. By integrating these elements, we aim to create computer vision algorithms that are more robust and adaptable to real-world applications, while being straightforward to develop and implement.

1. Metadata Distribution Alignment
   Visual semantics and the spatial relationships found in images are largely influenced by metadata, which includes attributes such as bounding box areas, rotations, positional coordinates, and background congruity. Addressing the existing disparity between domains, we focus on harmonizing the metadata distributions between synthetic and real image datasets. Using dataset engineering tools like data augmentation and filtering, we aim to make the metadata distribution of synthetic images closely resemble that of real images. This approach is proving vital in enhancing the algorithm's ability to generalize across

various domains.

2. Feature Space Distribution Alignment
   Moving to Feature Space Distribution Alignment, our goal is to align feature space representations that are often based on pre-trained models. We examine a range of feature extractors, including CLIP, Data2Vec, Dino, YOLO, and Faster R-CNN, to harmonize the feature space distribution between synthetic and real datasets. We are particularly focused on analyzing how different feature extraction techniques affect the alignment process, exploring both comprehensive and embedded feature sets. By methodically benchmarking these methods, we identify the strengths of each in closing the gap between domains, consequently improving generalization capabilities.

3. Sample-wise Correlation Analysis
   In addition to just achieving alignment, it is crucial to deeply understand the effectiveness of our alignment strategies. We undertake a detailed evaluation focused on the features and metadata of datasets, shedding light on the correlation between feature space closeness and detailed performance metrics. This aspect of evaluation provides clear insights into the capabilities and limitations of focusing on feature attributes, analyzed at different levels - from bounding box details to broader image-level diagnostics. These thorough analyses are vital in guiding the direction of future dataset engineering efforts, helping to pinpoint areas for improvement and innovation.

4. Dataset Evaluation During the previous alignment and correlation analysis pipelines, we methodically benchmark metadata and feature representations across diverse datasets. A good side job would be to devise a streamlined method to evaluate the dataset's quality and reliability, comparing metadata and feature distributions with those observed in validated real-world datasets. This strategy is geared not only towards gaining a finer understanding of dataset attributes but also toward promoting informed and strategic choices during the early phases of model training. This endeavor reflects our dedication to improving the initial stages of data analysis, thereby advancing the overall efficacy and progression of machine learning approaches.

3.2 Motivation

In this section, we aim to provide a deeper understanding of the methodologies we've outlined earlier. Our objective is to convey the reasoning that guided our choice of approaches and why we anticipate them to potentially be successful. This discussion isn't just about highlighting what we included but also shedding light on the paths we chose not to explore in this study. We

feel that understanding the reasons behind the exclusion of certain methods is equally important to comprehend the full scope of our research strategy. We aspire to offer a clear and balanced view, enabling readers to grasp the essence of our methodological framework in its entirety.

3.2.1 Motivation for Aligning Synthetic and Real Data

In the realm of deep computer vision, algorithms are inherently designed to grasp the data distribution present in the training dataset. Thus, aligning the distributions of synthetic and real datasets can significantly enhance the generalizability of these algorithms across diverse domains and refine performance on real-world data. However, it's imperative to understand that alignment does not equate to rendering synthetic images wholly realistic. This is mainly because computer vision models perceive images distinctively from humans. Moreover, generating ultra-realistic images is both computationally intensive and time-consuming, given the intricacies of physical simulations and comprehensive ray tracing. Additionally, domain generalization issues, prevalent even among real datasets, further complicate the alignment endeavor. Examples of this phenomenon can be found in related works chapter 2.2.1, as well as in experiments chapter 5.2.1, where naively sampled synthetic datasets lead to very poor generalization results on real datasets.

3.2.2 Complementing the Logic of Distribution Alignment

Historically, the consensus has been straightforward: if the distributions of synthetic and real datasets align, then the metadata attributes and feature space representations would naturally align as well. However, achieving explicit distribution alignment is extremely challenging. In this context, we explore the promise of utilizing metadata and features as pivotal anchor points to shape data distribution. The thrust of our experiments focuses on identifying metadata attributes that encapsulate a significant amount of information about dataset distribution, to minimize the disparity between synthetic and real datasets.

But why emphasize metadata and feature alignment? Our observations suggest that metadata serves as efficient abstractions, encapsulating the essential information derived from an image, even if its nuances might be elusive to human comprehension. It stands to reason that certain metadata can profoundly reflect the underlying distributions. Aligning them, thus, becomes a logical step. On the topic of features, it is widely accepted that feature extractors in deep learning models extract comprehensive information about the images they process. Even if these features aren't always intuitive to human analysts, their importance cannot be understated, especially given that many deep learning models are predominantly feature-based. A substantial

portion of these models is dedicated to feature extraction, underscoring the necessity for the extracted features to be information-rich. As such, aligning these features becomes crucial in steering the underlying distribution in the desired direction. The metadata we studied in this work is listed in sections 4.6.1 and 4.6.2, and the feature space representations we study are described in sections 4.7 and 5.3.1.

3.2.3 Experimental Methodology

Our methodological design is thorough and is centred around assessing the influence of various metadata attributes and feature space representations on computer vision algorithms. This data-centric approach encompasses both quantitative and qualitative evaluations. We utilize an array of metrics to assess the efficiency of computer vision models, particularly when trained on synthetic datasets with varying levels of metadata and feature alignment.

By holding other parameters constant and modulating the alignment degree, we are equipped to ascertain which metadata and features most significantly impact the underlying data distribution. This offers a foundation for proposing guidelines on which attributes should be prioritized for alignment and which may not be of paramount importance. To ensure a robust and unbiased understanding, benchmarks will be conducted across a spectrum of deep learning algorithms, datasets, and diverse objects of interest. This comprehensive evaluation strategy aims to furnish a well-rounded insight into the intricacies of metadata and feature alignment in the realm of computer vision. A high-level benchmark pipeline illustration can be found in figure 4.1.

A more concrete example can be found in the experiment chapter in section 5.2.1, where we benchmark one specific metadata attribute on datasets: the bounding-box area, which is essentially the size of the objects of interest in images in the context of object detection. In particular, we propose 3 different versions of the dataset, in which the alignment degrees in bounding-box area distribution are highly different, but the other configuration parameters, such as light, and shapes of the objects in the generation pipeline, are fixed in all three datasets. For benchmarking those three datasets, we train the same set of computer vision algorithms on those three datasets separately and evaluate the trained models on the real test dataset. We then check the correlation between alignment degrees and model performances. Only if there is a strong correlation across different datasets and computer vision algorithms, we would conclude that this metadata is an important attribute of synthetic datasets and its distribution should be aligned.

3.2.4 Deep Generative Model: A Viable Solution?

The idea of employing Generative Adversarial Networks (GANs) or similar deep learning methodologies to produce synthetic images that replicate real data distribution is tempting. GANs have indeed showcased exceptional capabilities in image synthesis. However, when scrutinized in the context of our study, they present certain limitations.

On a theoretical plane, GANs can be fine-tuned to generate images that align closely with the distribution of genuine images. However, a significant impediment with Deep Generative Models is their lack of precision in controlling the attributes of the images they generate. Such granular control becomes critical, especially when iterative processes are in play. If a deficiency is identified within a dataset, producing additional images tailored to that niche often becomes a challenge with these models. This limitation persists even in light of recent progress in prompt engineering, particularly with large-scale diffusion models.

Further complicating the prospect of deep generative models is the challenge of annotations. Many of these models do not incorporate modules to embed or predict annotations in the synthesized images. The subsequent effort required to annotate these images can be considerable, thereby raising concerns about the cost-effectiveness of using synthetic data. While tools like the 'segment-anything' model from Matter may offer a solution, they often fall short in providing intricate details such as depth maps or lighting conditions, critical aspects for our research considerations.

Chapter 4

Methodology

In this chapter, we set the stage by laying out the specific settings associated with the problem under investigation. This entails a formal definition of the task, emphasizing the elements we aspire to optimize. A formal definition is needed not only for the formality but also for generalizing the two different problem contexts we studied in the following chapters since they follow the same problem structure and can be viewed as different instantiations of the problem defined in 4.1. The first problem is to generate the best synthetic datasets given the real dataset, which is thoroughly discussed in the previous chapters. The second problem context is to modify the train datasets given a distribution shift in the test dataset when both train and test datasets could be real datasets, which is explained in sections 5.4.4 and 4.7. The following sections will detail the methodologies employed, underpinned by the rationale guiding our choices. Furthermore, the chapter underscores a systematic strategy tailored for the alignment of synthetic and real images. The motivation and rigorous analysis supporting this strategy are covered herein, with empirical findings reinforcing its validity to be presented in subsequent chapters.

4.1 Formal Problem Statement and Definitions

Let $\mathcal{D}$ be the distribution of data samples and let $X \sim \mathcal{D}$ be a set of data samples drawn from distribution $\mathcal{D}$, where $|X| \ll N$ with N being a certain constant. Let $\mathcal{F} : X \sim \mathcal{D} \mapsto \mathbb{O}$ be the function that takes a single data point $x \in X \sim \mathcal{D}$ as input and maps it to the output space $\mathbb{O} \in \mathbb{R}^d$. Further, suppose $\Psi(\cdot) : X \times \mathbb{O} \mapsto \mathbb{R}$ is the evaluation metric that takes a single data point $x \in X \sim \mathcal{D}$ and $o \in \mathbb{O}$ as an inputs and output a real value $r \in \mathbb{R}$ for evaluating the outcomes of the function $\mathcal{F}$ based on this data point x. Let $\mathcal{F}^*_{X'}$ denote the function F that optimizes $\Psi(\cdot)$ over an arbitrary dataset X', i.e. $F^*_{X'} = \text{argmax}_f \, \mathbb{E}_{X'}[\Psi(X', F(X'))]$. Now based on those definitions and

notations, our problem setting can be formalized as follows:

Definition 4.1 *Given dataset $X \sim \mathcal{D}$ sampled from implicit distribution $\mathcal{D}$, $|X| \ll N$, a function $\mathcal{F} : X \sim \mathcal{D} \mapsto \mathbb{O}$ and an evaluation metric $\Psi(\cdot) : X \sim \mathcal{D} \times \mathbb{O} \mapsto \mathbb{R}$, find $\widetilde{\mathcal{D}}^*$, such that:*

$$\widetilde{\mathcal{D}}^* = \text{argmax}_{\widetilde{\mathcal{D}}} \, \mathbb{E}_{X \sim \mathcal{D}}[\Psi(X, F^*_{\widetilde{X} \sim \widetilde{D}}(X))]$$

where $F^*_{\widetilde{X}} = \text{argmax}_f \, \mathbb{E}_{\widetilde{X} \sim \widetilde{\mathcal{D}}}[\Psi(\widetilde{X}, F(\widetilde{X}))]$.

Instantiations

In terms of synthetic and real datasets/distributions and under the context defined in 3.1, the target distribution D is viewed as the implicit real data distribution, $X \sim D$ as the real dataset we have with limited size, $\widetilde{X} \sim \widetilde{D}$ as the generated synthetic dataset and $\widetilde{D}$ as the implicit synthetic distribution. The problem reduces to finding the best synthetic datasets $\widetilde{X}$ given the real dataset X. And the best synthetic dataset $\widetilde{X}$ is defined as the one that maximizes 4.1, i.e. given F which is a deep learning algorithm, $\widetilde{X}$ should be the synthetic dataset that maximizes the metric function Ψ over the real dataset X when F is optimized(trained, under the context of deep learning) over $\widetilde{X}$. Note that X is used as a test dataset here, and Ψ is the evaluation metric for the deep learning model on the test dataset. In the context of supervised learning, when F is an image classification or object detection algorithm that takes as input a sample $x \in X$ and outputs the prediction $o \in \mathbb{O}$, Ψ is typically the Accuracy or MAP [14] metric. In the context of unsupervised learning, when F is for instance a clustering algorithm that takes x and outputs o as the assigned cluster to x, which is however not covered in this work, Ψ is typically the silhouette score used for clustering evaluation [41].

Of course, the benefit of generalized definition is that it gives us wider room for interpretation, not necessarily restricted to synthetic and real datasets. Another interpretation of 4.1 could be found in 5.4.4, where we interpret D as the implicit test data distribution in the COCO dataset, and $\widetilde{D}$ as the implicit train data distribution. The problem there becomes finding the best-modified train dataset $\widetilde{X}$ from the original train dataset given the test dataset X.

4.2 Limitations

As we already mentioned in the overview chapter, it is extremely hard to optimize over a general distribution $\widetilde{\mathcal{D}}$. The standard trick for optimizing that is using parameterization. Let $\mathcal{P}$ be the parameter space and Φ be a parameterized distribution over the same sample space as $\widetilde{\mathcal{D}}$. Then we assume

$$\widetilde{\mathcal{D}} = \Phi(\Theta)$$

for some certain parameters $\Theta \in P$, with Φ fixed. Then the optimization problem now becomes to :

$$\Phi^*, \Theta^* = \text{argmax}_{\Theta \in \mathcal{P}, \Phi} \mathbb{E}_{X \sim \mathcal{D}}[\Psi(X, F^*_{\widetilde{X} \sim \Phi(\Theta)}(X))]$$

In terms of synthetic data generation, we could view the parameter space $\mathcal{P}$ as the space of all possible configurations for the synthetic data generation pipeline, and the parameterized distribution Φ is implicitly defined by the implementation of the pipeline and the sampling strategies. Even if the pseudo-distribution $\widetilde{\mathcal{D}}$ can be parameterized, the optimization problem is still extremely hard, since it is almost impossible to parameterize the given distribution $\mathcal{D}$ in a meaningful way generally. Further, the optimization problem to obtain $F^*_{X \sim D}$ and $F^*_{\widetilde{X} \sim \widetilde{D}}$ is usually highly non-convex and difficult to solve. Therefore, to obtain meaningful results, we need to make approximations and assumptions.

4.3 Approximations

To optimize the utility function in actual experimental settings, we make the following approximation:

1. $\mathbb{E}_{X \sim \mathcal{D}}[\Psi(X, F(X))]$ would be approximated empirically with the sampled dataset $\hat{X} \sim \mathcal{D}$ with $|\hat{X}| = n$ and approximate with $\mathbb{E}_{X \sim \mathcal{D}}[\Psi(X, F(X))] \approx \frac{1}{n} \sum_{\hat{x} \in \hat{X}} \Psi(\hat{x}, F(\hat{x}))$
2. $\mathcal{F}^*_X = \text{argmax}_f \mathbb{E}_{X \sim \mathcal{D}}[\Psi(X, F(X))]$ would be approximated using an optimizer ϑ, which guarantees to converge in a local maximum, e.g. SGD [42] or Adam [43]. We would assume that $\mathcal{F}^*_X \approx \mathcal{F}^\vartheta_X$, where $\mathcal{F}^\vartheta_X$ is the F obtained by optimizing $\mathbb{E}_{X \sim \mathcal{D}}[\Psi(X, F(X))]$ using ϑ.
3. The pseudo-distribution $\widetilde{\mathcal{D}}$ can be parameterized with $\widetilde{\mathcal{D}} = \Phi(\Theta)$

Putting it all together, we change the optimization problem defined in Definition 4.1 to the following approximated optimization problem:

Definition 4.2 *Given dataset $X \sim \mathcal{D}$ sampled from implicit distribution $\mathcal{D}$, a function $\mathcal{F} : X \mapsto \mathbb{O}$ and an evaluation metric $\Psi(\cdot) : X \times \mathbb{O} \mapsto \mathbb{R}$, find $\Phi^*, \Theta^* \in \mathcal{P}$ such that $\mathcal{P}$ is the parameter space, $\Phi(\mathcal{P})$ forms a probability distribution over certain sample space, and*

$$\Phi^*, \Theta^* = \text{argmax}_{\Theta, \Phi} \frac{1}{|X|} \sum_{x \in X} \Psi(x, F^\vartheta_{\widetilde{X} \sim \Phi(\Theta)}(x))$$

where $F^{\vartheta}_{\widetilde{X}\sim\Phi(\Theta)}$ *is the optimized solution obtained by using optimizer* ϑ*.*

In the scope of this work, the approximated value for performance metric $\Psi(X, F) = \frac{1}{|X|}\sum_{x\in X}\Psi(x, F^{\vartheta}_{\widetilde{X}\sim\Phi(\Theta)}(x))$ is typically the value of performance metric computed on the real test dataset with the pre-defined metric function Ψ using the model $F_{\widetilde{X}\sim\Phi(\Theta)}$ trained on $\widetilde{X}$.

4.4 Optimization Approaches

Given a parameterized distribution Φ, parameter space $\mathcal{P}$ and an optimizer ϑ, Definition 4.2 gives us a computable utility function. However, this utility function is not end-to-end differentiable, which disallows us from using numerical optimization algorithms which are typically gradient-based. Further, in practise, it is very costly to optimize $F^{\vartheta}_{\widetilde{X}\sim\Phi(\Theta)}$ at each step. Typically, when F is a deep learning algorithm, the optimizer ϑ would take a few hours, even a few days to converge to a local minima/maxima. Therefore, we have to find efficient heuristic approaches. In this section, we will propose two different heuristic approaches: Bottom-Up and Top-Down approaches, and evaluate their advantages and limitations. The bottom-up approach starts with matching the target data distribution D and producing realistic samples by directly learning D from the dataset $X \sim D$. Top-down approaches optimize its $\widetilde{D}$ by an iterative process using the evaluation metric Ψ. At each step, the approach evaluates the currently generated pseudo-distribution $\widetilde{D}$, then improves it in the next step. In the end, we will present our approach.

4.4.1 Bottom-Up Approach

The most direct way would be to align $\Phi(\Theta)$ with target distribution $\mathcal{D}$, i.e. $\Phi(\Theta) \approx \mathcal{D}$. Further, the search space for $\Phi(\Theta)$ is too big and typically leaves no trace for us to directly optimize over the entire space. Therefore, an empirical approach would be to first select a relatively simple parameterized distribution space $\mathcal{S}$, where each distribution $\Phi \in \mathcal{S}$ can be fully described by its Parameter Space $\mathcal{P}$. The choice of $\mathcal{S}$ is typically based on strong prior and observation on the target distribution $\mathcal{D}$. By fixing $\Phi \in \mathcal{S}$, we can optimize the parameters $\Theta \in \mathcal{P}$ by heuristic approaches like grid search, and forward sampling that are typically used in hyperparameter search for machine learning algorithms [44]. For evaluating the alignment, we can use some empirical metric for measuring distances in high-dimensional distributions, like approximated KL-Divergence [45], kernel density estimation [46].
The biggest advantage of this approach is that it arises naturally and is very straightforward. Aligning the pseudo distribution $\widetilde{D}$ with the target distribution D is indeed the ultimate solution to the optimization problem in the definition 4.1, because it is easy to see that if $\widetilde{D} = D$, 4.1 must be optimized. However, it has the following limitations:

1. By pre-selecting a certain parameterised distribution S, we introduce a strong bias a prior, which almost solely determines the optimized solution. Further, there is almost no guarantee that the pre-selected parameterised distribution family contains any distribution that is even close to the implicit target distribution $\mathcal{D}$ underneath. We could find the optimised parameters and the evaluation metric $\Psi(\cdot)$ is still bad.
2. The heuristic approaches could be very time-consuming. To evaluate the alignment, one would need to sample $\widetilde{X} \sim \widetilde{\mathcal{D}}$, and use a high-dimensional distribution measure to compare $\widetilde{X}$ and $X \sim \mathcal{D}$. In high-dimensional cases, e.g. with image samples, a single iteration of sampling and computing would already be time-costly, and a grid search/forward-sampling/backward-sampling would take numerous iterations to complete.
3. In a practical data generation pipeline using a renderer, there is almost no explicit control over the distribution of the generated samples. There is no possibility to first choose a certain explicit distribution from mathematical space, pass it to the render and get the desired samples generated from this input distribution.

Using deep generative models, such as VAE [20], GAN [21] and Diffusion Model [47] to reconstruct samples from $\mathcal{D}$ given the dataset $X \sim \mathcal{D}$ can also be viewed as examples of such Bottom-Up Approach because they aim to directly learn the generative target distribution. From the problem definition 4.2, we know that an optimal solution can be achieved by alignment(if we can align $\widetilde{D}$ with D, we can achieve the global optimum), though it might not necessarily be the only optimal solution. We call it the bottom-up approach because it starts with the basic fundamental characteristics of the problem: the pseudo-distribution and the target distribution. However, a direct alignment in distribution is practically infeasible, because the target $\mathcal{D}$ can be arbitrarily complex, and we do not have access to the target distribution itself. Even in the case of deep generative models, where the learned distribution may also be implicit and even arbitrarily complex, the alignments often turn out to be insufficient, as shown in [48]. Further, training a deep generative model requires a lot of data samples, which is infeasible in our setting, as we assume the dataset X is small, i.e. $|X| \ll N$.

4.4.2 Top-Down Approach

A second approach would be to not make any assumptions on the distribution underneath and optimize the parameters, but use an iterative approach based on the observations on $\Psi(x, F^{\vartheta}_{\widetilde{X}\sim\Phi(\Theta)}(x))$, $x \in X \sim \mathcal{D}$. Suppose we have an initial sample generation pipeline that follows certain distribution $\Phi_0(\Theta_0)$ and the generated dataset is $\widetilde{X_0}$. Then we follow a heuristic iterative

optimization approach, i.e. based on $\Psi(x, F^{\vartheta}_{\widetilde{X_0}\sim\Phi_0(\Theta_0)}(x))$, we find the underperforming samples $x \in X \sim D$, denoted as $\hat{X}_0$. Then for evolving into the next iteration, we sample in a certain set of Φ and parameter space $\mathcal{P}$ to generate $\widetilde{X_0^*} \approx \hat{X}_0$. Then we evolve the dataset with $\widetilde{X_1} = \widetilde{X_0} \cup \widetilde{X_0^*}$. We call it a top-down approach because it uses the top-layer information from the optimization problem: the metric function $\Psi(\cdot)$ itself. By finding the "blind spots" from $\Psi(\cdot)$, the distribution underneath is implicitly improved by keep adding new samples into the generated dataset. The process would stop when $\Psi(x, F^{\vartheta}_{\widetilde{X_0}\sim\Phi_0(\Theta_0)}(x))$ are above a certain threshold or when the differences in the last two iterations are negligible.
In the context of generating synthetic object detection datasets, the top-down approach would work as follows: first, generate a baseline dataset $\widetilde{X_0}$, then evaluate the object detection model F trained on $\widetilde{X_0}$ using e.g. the Average Precision Ψ, on the test real dataset. Then a subset of X, denoted as $\hat{X}_0$, is selected from X, where the model F performs poorly on this subset, i.e. the average precision on that subset, $\Psi(\hat{X}_0, F(\hat{X}_0)$ is below a certain threshold. Then another set of synthetic images $\widetilde{X_0^*}$ is generated based on $\hat{X}_0$, and merged into $\widetilde{X_0}$. The process then repeats itself. The limitations of this approach are the following:

1. The iteration could potentially be much more time-costly than the bottom-up approach since one would not only need to sample $\widetilde{X} \sim \widetilde{\mathcal{D}}$, but also fit $F^{\vartheta}_{\widetilde{X}}$. When F is a complicated algorithm, the fitting would be very time-consuming and dominate the overall time consumption in the iteration. In a real-world scenario where time is usually limited, we could potentially only proceed few iterations from the beginning.

2. The process is not easily generalizable since it is an iterative process based on one specific starting point $\widetilde{X_0}$ and one target dataset X. If we have a next target dataset $X\prime$ that is drawn from a different distribution, the whole process must be repeated and the previous iterations would give zero insight into the new iterations.

3. There is no guarantee that the iterative process will ever converge or meet the stopping threshold. Further, there is also no guarantee that the evaluation metric $\Psi(\cdot)$ would be monotonic increasing through the iterations. If $\Psi(\cdot)$ decreases after certain steps, it then becomes clueless whether the iterations should be kept proceeding or not.

4. It remains unclear how to sample the parameter space and distribution space to find a good $\widetilde{X_0^*} \approx \hat{X}_0$ within each iteration step. However, the quality of the dataset $\widetilde{X}$ in the end is almost solely dependent on the quality of each iteration step.

4.4.3 Our approach

Our innovative approach to minimizing the gap between the pseudo and target distributions combines the principles of both top-down and bottom-up methodologies, creating a synergistic framework that overcomes the limitations of these conventional strategies. It can be summarized as follows:

1. We capitalize on the strengths of bottom-up distribution alignment, while elegantly sidestepping the constraints of explicit parameter optimization. In our pursuit of aligning the pseudo and target distributions, we seek to identify a set of attributes that, when aligned, effectively reduce the distribution gap. Rather than pursuing explicit optimization, our approach leverages implicit alignment. We select these attributes as anchor points, allowing us to exert control over the underlying distributions. This method hinges on identifying and harnessing attributes with significant influence over the distribution alignment process. More details can be found in section 4.6.

2. To discern the ideal set of attributes for distribution alignment, we engage in extensive benchmarking. Through systematic experimentation with diverse datasets, each characterized by unique attributes, we evaluate the contributions of these attributes to the distribution alignment process. The goal is to uncover a set of attributes that holds relevance across a range of datasets, enabling a generalized approach to metadata distribution alignment. More details can be found in section 5.2.1.

3. Our optimization strategy harmonizes the virtues of both top-down and bottom-up approaches, offering a unified and potent solution. We initiate by creating an initial pseudo-distribution, $\widetilde{D_0}$, through preliminary attribute alignment in the data generation pipeline. This pseudo-distribution acts as a foundational platform for iterative refinement. Our iterative steps encompass the continuous enhancement of $\widetilde{D_0}$, guided by its proximity to the target distribution and insights gleaned from evaluation metrics. These iterations involve a dynamic interplay of filtering and augmentation techniques, contributing to the gradual and effective alignment of the pseudo-distribution. More details can be found in 4.3 and 5.4.4.

4. To assess the distribution gap between the pseudo and target distributions, we adopt a dual-pronged approach that harmonizes the benefits of both top-down and bottom-up perspectives. While evaluation metrics offer precision, they may also consume considerable time. Instead, we undertake an empirical exploration of the relationship between evaluation metrics and distribution distance measures. By thoughtfully combining these measures, we extract relevant information that em-

powers the optimization process. This streamlined evaluation process enhances the efficiency of distribution alignment efforts. More details can be found in 5.4.6.

An illustration of using our approach on synthetic distribution alignment is illustrated in figure 3.1.

4.5 Problem Settings

As mentioned in the previous chapters, we would restrict ourselves to a smaller subset of the optimization problem defined above in this project. It is harder to evaluate when the setting is too broad and not specific at all. More specifically, we would focus on object detection problems, which have a wide range of applications and great importance in real life. Therefore, the function $\mathcal{F} : X \sim \mathcal{D} \mapsto \mathbb{O}$ would be a deep object detection algorithm such as YOLO [49], Faster R-CNN [50] and FCOS [51], and the metric function $\Psi(\cdot) : X \sim \mathcal{D} \times \mathbb{O} \mapsto \mathbb{R}$ would be some widely used evaluation metrics for object detection algorithms such as MAP, AP50, AP75, bbox-iou etc [14]. Further, as described beforehand, our target distribution $\mathcal{D}$ would be interpreted as the real image sample distribution, and the pseudo distribution $\widetilde{\mathcal{D}}$ would be interpreted as the synthetic image sample distribution. Therefore, we could view the optimization problem with closing gaps between synthetic datasets and real datasets as a special case of the generalized optimization problem defined in 4.1. Such restrictions give us insights into how to approach the generalized optimization problems, and the generalization also empowers those approaches to be reused in other cases of the optimization problem 4.1, since the problems themselves would have the same inherent structures. Further, as described in the introduction chapter, we would proceed with a realistic setting. In real-world scenarios, synthetic datasets are usually introduced when the real dataset is very limited. Therefore, in our setting, we would also assume that we have no direct access to the target distribution $\mathcal{D}$ underneath. Instead, we only have access to a small dataset $X \sim \mathcal{D}$ with $|X| \ll N$, where typically $N \leq 100$.

4.6 Metadata Alignment for Underlying Distribution

To identify pivotal anchor points for the underlying distribution, we initially focus on the metadata alignment and benchmarking the results of the alignment as well as the performance of the models trained on those datasets, with/without the alignments. An illustration of the benchmark pipeline is shown in the figure 4.1.

Within the realm of object detection, both real datasets, represented as $X \sim \mathcal{D}$, and the synthetic one, $\widetilde{X} \sim \widetilde{\mathcal{D}}$, contain also annotations respectively. For

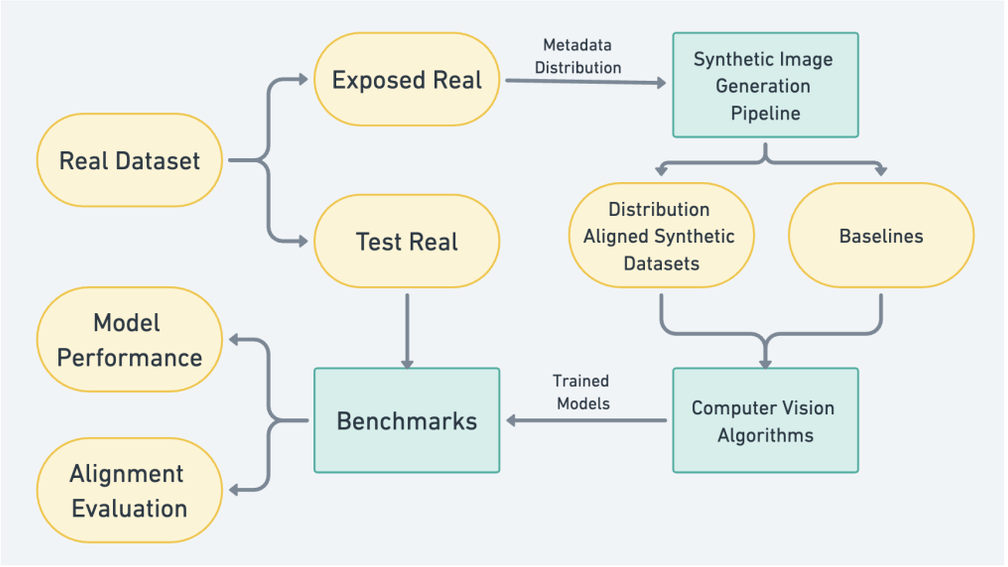


Figure 4.1: The real dataset is split into two parts, one is exposed and the other one is for testing. The exposed part feeds its metadata distribution to the image generation pipeline, which produces a few baseline datasets as well as metadata-aligned datasets, which are then used for training various deep learning algorithms. The models are then benchmarked as well as the effect of the alignment.

the sake of consistency and relevance, our focus remains primarily on bbox-format annotations, which are employed across object detection datasets, i.e. we would assume that the real datasets do not contain any other annotations than the bounding box annotations.

While we operate under the assumption of the absence of more detailed annotations on the real datasets, including segmentation information, the synthetic datasets provide a broader scope. Given that these datasets are crafted and rendered via our proprietary pipeline, they enable us to access the supplementary details. This includes the segmentation masks of objects, their distances to the camera, and more.

4.6.1 Exploring Basic Metadata

To begin, we examine the fundamental metadata that can be directly derived from the image samples along with their associated annotations, including attributes like image blurriness and brightness. Both real and synthetic datasets readily provide these attributes. Detailed methods to compute these characteristics are available in the appendix, primarily involving the use of filters.

Image-Level Metadata

Image-level metadata pertains to the inherent properties of the images themselves and represents the foundational attributes of any image. For benchmarking, we have identified key image-level metadata: image sharpness, image blur, image entropy, image brightness, and image contrast. Importantly, these attributes are computed without the need for bounding-box(bbox) annotations. However, this also signifies that these metrics lack object-detection relevant information, potentially rendering them less valuable for specific use cases.

Bounding Box (Bbox)-Level Metadata

Bbox-level metadata, on the other hand, relates to properties extracted from the bounding boxes as specified in the annotations. Contrasting with image-level metadata, these attributes are more granular but packed with information relevant to object detection. This relevance stems from the fact that advanced object detection algorithms heavily utilize annotation data when defining their loss functions. For benchmarking, we have zeroed in on several bbox-level metadata: bbox sharpness, bbox blur, bbox entropy, bbox brightness, bbox contrast, bbox area, and bbox aspect ratio. The methods to calculate sharpness, entropy, blur, and contrast mirror those employed for image-level metadata. Bbox area represents the space occupied by the bounding box, while bbox aspect ratio is determined by comparing the width to the height of the bounding box.

4.6.2 Advanced Metadatas in Object Detection

Traditional metadata offers utility in both real and synthetic datasets. These types of metadata, which can be categorized as ”basic”, are relatively simple to compute. A common example is the computation of entropy from pixel values. However, these calculations only provide a superficial understanding since they lack spatial contextual information such as the position or relational position of pixels. For instance, the reordering of pixels inside an image or bounding box (bbox) wouldn't change its entropy even though the semantic content might have drastically shifted. This highlights the limitations of relying solely on basic metadata. Consequently, there's a clear need to introduce and analyze advanced metadata which offers deeper insights. Notably, some of these sophisticated metadata types, such as background similarity, object occlusion, shape, and depth, rely on information from rendering engines. This dependency predominantly ties them to synthetic datasets. Nevertheless, this section will detail methods to both extract this information from renderers and estimate similar metrics for real-world datasets. Subsequent chapters will delve into the evaluation of these proposed methodologies.

Background Similarity

The concept of background similarity revolves around comparing an object to its immediate surroundings. The underlying hypothesis is straightforward: objects closely resembling their background can be more challenging for detection algorithms. To derive this metric, especially in synthetic datasets, we require a segmentation mask from the rendering engine. A typical bounding box might include both the object of interest and some of its surroundings, rendering it suboptimal for this specific task. The computation method is as follows:

1. Extract pixels on the boundary of objects, along with their adjacent background pixels.
2. Compute the cosine similarity between these extracted objects and background boundary vectors.
3. The concrete mathematical formulation is $\cos(v_{background}, v_{object})$, where $v_{object} = \bigoplus_{i,j \in O} p_{i,j}$, $\bigoplus$ being the vector concatenation function, O is the set of object boundary pixel coordinates that have neighbouring background pixels, p being pixel RGB vector and $v_{background} = \bigoplus_{i,j \in B} p_{i,j}$, where B is the coordinate set of a background corresponding to O.

To find all pixels on the boundaries of the objects and their neighbouring background pixels, we deploy the morphological dilation [52] with a square mask of size 3. Notably, the computation of background similarity is restricted to bbox-level because it necessitates individual object instance details.

Object Shape

Object shape is an integral feature that depicts the contour and form of an object, serving as one of its foundational characteristics. Primarily classified at the bounding box (bbox) level, it is imperative to preserve this metadata for analysis. To capture this information, we fetch the object's name directly from the rendering engine. Throughout the synthetic data generation pipeline, every distinct object is assigned a unique identifier. This identifier is subsequently registered within the renderer. During the image sample creation process, this identifier is stored in tandem with its corresponding bounding box.

Depth

Depth represents the spatial distance between an object and the camera. A principal consideration behind measuring depth stems from the understanding that objects farther from the camera are inherently more challenging to detect compared to proximate ones. Parallel to the mechanism employed for object shape, the renderer is capable of generating a depth map for every

image sample, along with its segmentation mask. By integrating information from both the depth map and the segmentation mask, the depth of each object encapsulated by a bounding box can be ascertained. It's pivotal to understand that an object's depth does not invariably correlate with the area of its bounding box. Such instances arise when the object is only partially visible due to intervening obstructions. Consequently, despite being situated closer to the camera, its bounding box might appear diminutive. This observation further informs our calculation of occlusion metrics for individual objects.

Occlusion

The occlusion of an object is defined as how much of the object is occluded by another object/obstacle that is between it and the camera. For, a chair could be occluded by a table and by other chairs near it. In those cases, the shape of the object can not be fully observed, which could be challenging for the object detection algorithm. However, there is no straightforward way to record the occlusion of the object, even with information from the renderer, because the segmentation mask does not contain information on how many pixels an object should spread over an image without occlusion. To that end, we would use the following insight:

The number of visible pixels of an object should be correlated with the depth and shape of the object if it is not occluded.

The insight comes from the fact that for objects with the same shape, the area shown in an image is linearly dependent on the square its distance to the camera, due to fundamental optical property, i.e. $a \sim \frac{1}{d^2}$, with a as the number of visible pixels of the object, and d as the depth of that object. This implies that the product of visible pixels and the squared depth should be a constant if there is no occlusion, i.e. $a \cdot d^2 \approx C$, C constant across all the objects, regardless of its depth to the camera because its area on the image is solely dependent on the depth when no occlusion exists. Therefore, an indicator for occlusion would be the case when the visible area of the object is surprisingly small compared to all other objects of roughly the same depth to the camera, in which case $a \cdot d^2 \ll C$. Further, the depth of an object to the camera is purely objective and irrelevant to occlusion, since it is directly obtained from the renderer. It encourages us to use $a \cdot d^2$ as an indicator for occlusion. More specifically, if for some objects, $a \cdot d^2 = \widetilde{C}$ is much smaller than other objects, we would know that the visible area of those objects is much smaller than it should be, which can only be explained by an occlusion. This object must be occluded by another object, and its visible area is therefore restricted. Further, $a \cdot d^2$ could also tell us how much of the object is occluded. More concretely, if we have the ground truth C, then

$\frac{a \cdot d^2}{C}$ should be roughly the percentage of the object visible from the camera, and the rest is occluded. To conclude, we would use $a \cdot d^2$ as the metric for occlusion. The lower the value, the more the occlusion. In particular, the correlation is linear, as emphasized beforehand.

Computing advanced metadatas for real dataset

As previously discussed, one of the main challenges faced in this domain pertains to the computation of advanced metadata on real datasets. Historically, the reliance has been on renderer-generated information for such computations. To address this limitation, we introduce a methodology to estimate these metadata in real-world datasets. Our approach hinges on leveraging a regressor or classifier, trained specifically to compute this metadata from real image samples. Notably, feature extractors play a pivotal role, as they often encapsulate data related to shapes, occlusion, and background contexts. Herein, we outline the steps involved in calculating this metadata for real datasets:

1. Begin by identifying a robust feature extractor, primed to extract intricate metadata details from synthetic samples.
2. Utilize the derived features to train a regressor or classifier. The training will be supervised using the metadata labels extracted from synthetic samples.
3. Implement cross-validation to assess the outputs of the regressor or classifier in tandem with the feature selection process. Should the outcomes not meet predefined thresholds, we advocate for a transition to an alternative feature extractor. This cycle of extraction, assessment, and replacement continues until satisfactory results are obtained. This iterative approach ensures an optimal feature extractor-regressor combination.
4. Finally, real image samples are fed into the finalized feature extractor. Subsequently, the trained regressor or classifier is employed to generate predictive metadata values.

The figure for illustrating the described pipeline is shown in figure 4.2.

An in-depth evaluation of this proposed methodology, along with its efficacy and nuances, will be expounded upon in the ”Experiments” chapter, under section 5.3.

4.6.3 Aligning metadata distributions

In line with the methodology highlighted in 4.4.3, our objective is to align the metadata, leveraging them as reference points for the underlying data

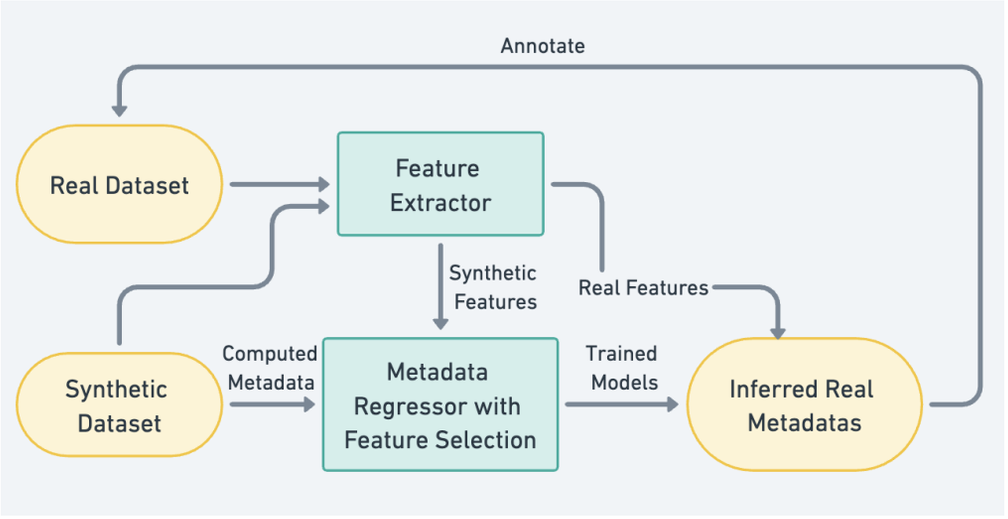


Figure 4.2: For predicting advanced metadata values on a real dataset, we compute the desired metadata from the synthetic dataset, fit a regressor using the synthetic features labeled with the metadata values, and then predict the values on a real dataset with the real features.

distribution. Aligning the metadata distribution presents fewer challenges compared to data distribution due to its one-dimensional nature and explicit capturability. Here, we detail a heuristic method for metadata alignment:

1. For all samples in the real dataset X, compute metadata values. These are introduced to the synthetic image generation pipeline and subsequently adjusted to fit a parameterized model, such as a Gaussian Mixture Model (GMM) [53].
2. Identify a modifiable set of configs within the image generation pipeline.
3. Execute a grid search over these configs for synthetic image generation. Each config yields a synthetic dataset generated using this config paired with its respective metadata values, computed from renderer information.
4. To approximate the real metadata distribution, we employ optimal combinations derived from the grid search. This involves the selection of a config subset, synthesis of samples using these configs, and their consolidation into an aggregated dataset. The resultant dataset is projected as the closest representation of the real metadata distribution.

Shifting the focus to the strategy for configuration subset selection, the idea is to fit the real metadata distribution with some clustering algorithms and then find the best config from the grid search for each cluster. The clustering algorithm could be K-Means [54], Gaussian Mixture Models [53], and the

number of clusters is automatically chosen by the silhouette score [41], which is widely used for determining the proper number of clusters. The motivation for using cluster algorithms is explained in the following paragraphs. To select the best config, we feed the sampled config to the image generator which produces a small set of generated synthetic image samples, with metadata labeled. Then we compare the synthetic metadata distributions of these sampled configs with the cluster parameters and find out the best config for each cluster. The figure for illustrating the described metadata alignment pipeline in synthetic image generation is shown in figure 4.3.

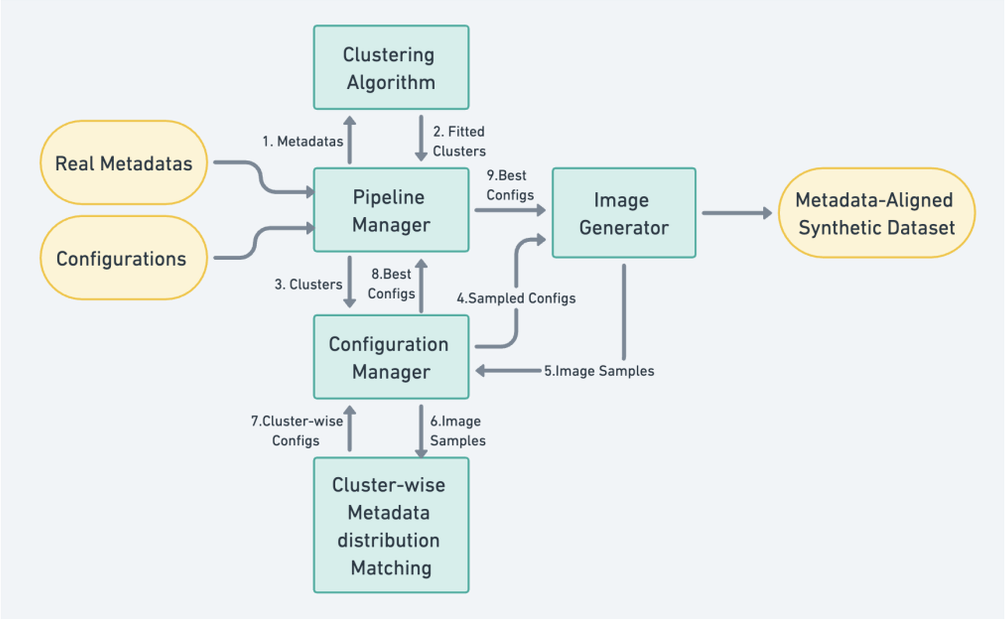


Figure 4.3: The metadata alignment pipeline consists of the following steps: Firstly, the pipeline manager feeds the real metadata to a clustering algorithm, and receives the parameters of the clusters. Secondly, it sends the parameters to the config manager, which selects the best config for each cluster by sampling configs and synthetic samples that are best matched with the metadata clusters. In the end, the subset of best configs is fed to the image generator and a final dataset is produced.

The first motivation behind this approach is that a standard clustering algorithm can almost fit any 1-dimensional distribution and it is easier to select the best config for each simple cluster than directly select the subset of configs. Another motivation stems from our observation: synthetic image metadata distribution consistently exhibits a Gaussian trend when a single fixed config is applied. Given the efficiency of GMM in capturing one-dimensional metadata distribution, we can emulate it using a configuration subset; each configuration represents a Gaussian component of the GMM.

Testing every configuration combination isn't practical due to the steep computational overhead. However, understanding that each GMM cluster is defined by its mean and variance, and given the Gaussian nature of the synthetic image metadata distribution under a singular configuration, the ideal configuration can be pinpointed based on proximity in mean and variance to the actual metadata distribution. After these configurations are finalized, the volume of samples generated from each becomes pivotal. GMM offers specific weights for its components, indicating the proportion of samples linked to each. These weights thus guide the determination of samples per configuration.

4.7 Feature Space Alignment

Aside from metadata, sample features derived from feature extractors serve as an alternative set of anchor points. Features often encapsulate both overt and covert data-specific details. This essentially gives insight into the model's perspective of the samples. Notably, deep learning algorithms might discern information in samples that transcend mere comprehensible metadata. One must recognize that the way a model perceives samples could be substantially different from human interpretation. This aspect has been discussed in depth within the domain of deep learning interpretability and explainability [55]. This insight necessitates aligning the feature distribution of the target data set, denoted as $\mathcal{D}$, with the surrogate distribution, $\widetilde{\mathcal{D}}$.

Another merit of leveraging features lies in their potential for an enhanced signal-to-noise ratio compared to raw image samples. This enhancement stems from the fact that the details get refined through the feature extractors. Several renowned deep learning architectures, including YOLO and Faster R-CNN [50], essentially consist of a feature-extraction backbone, complemented by simplistic regressor heads, usually with fully connected layers or 1D convolution layers. Given this structure, one can infer that the features must harbor crucial data, as the efficacy of these algorithms hinges on it. Nevertheless, it's imperative to judiciously choose an apt feature extractor and an evaluation scheme; not every extractor might align with the intended application. To facilitate this, we propose our methodology for evaluating various feature extractors. Another potential hurdle is aligning the high-dimensional distribution, a challenge we address in this segment. Subsequent chapters will shed light on the empirical results.

Feature Extractors Selection

The field of machine learning offers a plethora of feature extractors, each catering to distinct requirements. Broadly, these extractors can be classified into two categories:

1. **General vs. Specialized**: General feature extractors such as the ResNet Backbone and the RetinaNet Backbone are designed to be versatile, suitable for a wide range of applications, especially in the domains of transfer learning and representation-based learning [56]. In contrast, specialized feature extractors, exemplified by the backbones used in Faster R-CNN and YOLO, are tailored specifically for niches such as object detection.
2. **Pre-trained vs. Custom-trained**: The initial weight configurations of feature extractors are pivotal to their performance. Pre-trained weights, commonly derived from expansive datasets like ImageNet or methodologies like contrastive learning, offer a robust starting point. Alternatively, custom-trained weights, fine-tuned on specific tasks or datasets like $X \sim \mathcal{D}$ for object detection, can also be employed.

It's imperative to thoroughly benchmark these feature extractors to discern their appropriateness for specific applications.

Benchmarking Methodologies

The intricacy of feature and feature extractor selection predominantly hinges on establishing benchmarking and evaluation mechanisms. For our purpose, we need to select the best subset of feature extractors which contain the most information regarding object detection. To address this, we introduce the following evaluation technique:

1. **Metadata Fitting**: An informative feature extractor should extract enough information regarding the attributes from the objects of interest, e.g. shape, size, its similarity to the background etc., which are effectively the advanced metadatas we collected in the section 4.6.2. The evaluation of feature extractors is therefore based on supervised learning. Assuming access to a comprehensive set of informative metadata attributes (e.g., background similarity, object shape, contrast), the underlying expectation is that the features produced by chosen feature extractors should encapsulate these attributes effectively. To evaluate, a regressor or classifier is appended to these features, trained to predict the aforementioned metadata, with the features remaining static during this training. For regression, we use linear regression [57] as the regressor and for classification, we use logistic regression [57]. For feature selection, we use Random Forest [58]. Performance metrics derived from this model offer insights into feature quality. The computation methodologies for these metadata attributes are detailed in the preceding sections.

The simple illustration of this benchmark is shown in the figure 4.4.

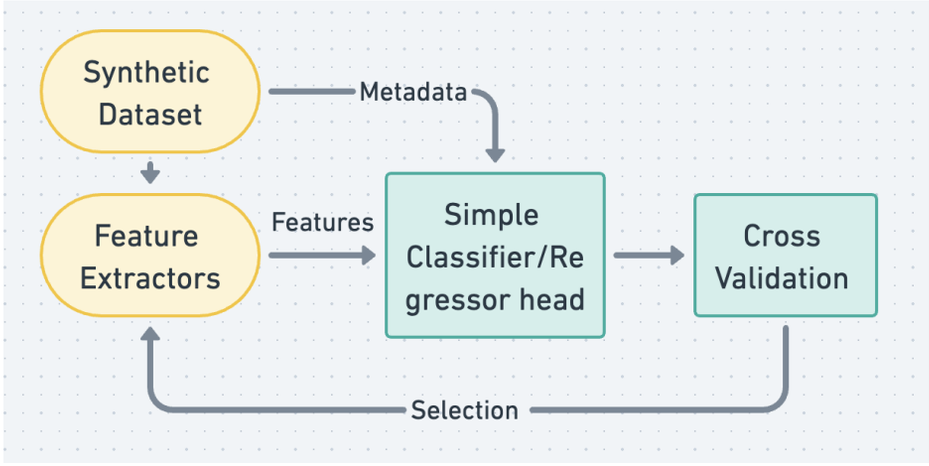


Figure 4.4: In the benchmark, the feature extractors produce features from synthetic samples. The features, along with the annotated metadata with the features, are fed into a simple classifier/regressor head, which uses cross-validation to select the best feature extractors.

Aligning Feature Distributions

Aligning high-dimensional feature distributions poses a greater challenge compared to metadata alignment due to computational cost and complexity. Direct alignment during sample generation within the pipeline remains impractical. Leveraging non-parametric methods, such as kernel density estimation or K-nearest neighbours, can aid in addressing this challenge. Consequently, we introduce two post-processing strategies to align these distributions:

1. **Feature Filtering**: This entails the removal of synthetic samples that deviate significantly from the real feature distributions. Non-parametric techniques, namely Kernel Density Estimation and K-Nearest Neighbours, facilitate the computation of likelihoods for each sample. Samples with the lowest likelihoods can be excluded entirely or their bounding boxes can be masked. An essential component of this method is setting an optimal likelihood threshold, which is established through the automatic elbow method [59].

2. **Feature Augmentation**: The objective here is to enrich the synthetic dataset, $\widetilde{X} \sim \widetilde{\mathcal{D}}$, with samples closely resembling real feature distributions. Generating such samples directly can be daunting. To circumvent this, we suggest a two-step strategy:

    a) From the generated synthetic dataset, identify a subset with high likelihood scores, employing non-parametric models combined with thresholding techniques.

b) Generate more samples from this subset using the image generation pipeline for augmenting the synthetic dataset.

Figure 4.5 illustrates the feature alignment pipeline as post-processing steps on the existing datasets. Note that this technique can be used on any dataset, not only synthetic ones. An example of using feature distribution alignment between real datasets can be found in section 5.4.4.

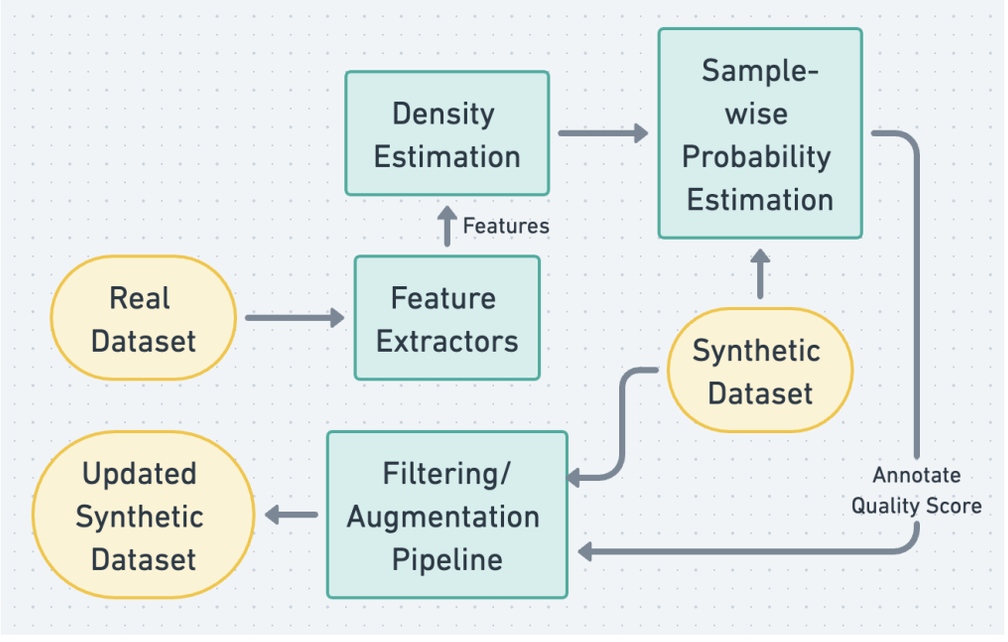


Figure 4.5: An overview of the feature alignment pipeline. The selected feature extractors extract features from the real dataset, fit the distribution with a non-parametric density estimator, then use it to evaluate each sample from the existing synthetic dataset, assign a quality score to it computed as the probability of its being drawn from the real feature distribution, which is then used for further filtering/augmenting to produce the updated dataset.

Chapter 5

Experiments

This chapter describes the experimental framework, benchmarks, configurations, and results performed as part of this work. We introduce the methods detailed in the methodology chapter and subject them to rigorous evaluation.

Reverting to our problem definitions detailed in 4.1, our experiments focus on object detection algorithms using deep learning models. The target distribution $\mathcal{D}$ signifies the real data distribution, while the pseudo distribution $\widetilde{\mathcal{D}}$ represents the synthetic data distribution, $X \sim D$ being the real dataset. The function $\mathcal{F} : X \mapsto \mathbb{O}$ is a deep object detection algorithm. In our study, we would choose algorithms such as YOLO [49], Faster R-CNN[50], and FCOS[51], which have established their reputation as state-of-the-art object detection algorithms for real-world applications, as our candidates for F. As for our evaluation metric $\Psi(\cdot) : X \times \mathbb{O} \mapsto \mathbb{R}$, our primary choice is the MAP (Mean Average Precision) [14], the most widely used metric for evaluating object detection algorithms. However, certain scenarios necessitate the use of auxiliary metrics, including AP50, and AP75 [14], among others, to gain more insights into the results. For the synthesis of data, Blender serves as our rendering tool, supplemented by Blenderproc for scripting, to streamline our rendering pipelines.

5.1 Settings

Our experimental setting operates on the assumption that our real dataset is restricted in terms of sample size. Nonetheless, for empirical study and an unbiased performance assessment, a bigger dataset becomes indispensable, as it mitigates the potential for significant variance in results. Consequently, while we employ a comprehensive real dataset, only a minuscule fraction is made accessible to the training apparatus.

The training pipeline adopts the conventional train/validation/test data

split. Additionally, to distinguish between the datasets, data labels are split into synthetic and real. Emphasizing the use of real data exclusively for validation and testing purposes ensures that only synthetic data contributes to the training partition. Furthermore, model selection and the termination criterion are dependent on metric performance on the validation subset. This validation portion comprises a handful of real samples (approximately 50), establishing a controlled environment. Conversely, the test subset is significantly expansive, ensuring a comprehensive and reliable approximation of the evaluation metric across the real sample distribution.

Throughout the study, we have used primarily three real datasets: the real chair dataset, the real table dataset and the real bag dataset. Each of them contains roughly 500 samples, where around 50-60 of them are randomly selected and used in the validation set, and the rest is used in the test dataset. For each real dataset, we generated many versions of synthetic datasets for different purposes. All of them have around 2,500 samples, where the train, validation and test set are randomly split with probability 0.7, 0.15 and 0.15. Although we also have the synthetic validation and test splits, we only use them for monitoring the training and sanity check. They are never used in the evaluation.

5.2 Basic Metadata

Building upon the preceding discussions describing our methodology in metadata alignment 4.6, this section explores and benchmarks the results in aligning metadata. We first analyze basic metadata types before advancing to more complex variants, systematically presenting our evaluative findings.

Our evaluation follows a two-step strategy: assessing model performance and evaluating the alignment. To judge model performance, we use the benchmark described in the prior ”Settings” section and illustrated in figure 4.1. This involves training models on a synthetic dataset, validating them using a subset of the real dataset, and finally testing them on the entire real dataset, and evaluating both the model performance and the alignment. The evaluation metrics are then computed based on the results from this test dataset.

In terms of alignment, we utilize the 1-dimensional Bhattacharyya distance [60] to measure the overlap between two 1-dimensional distributions, where a smaller Bhattacharyya distance indicates a higher degree of alignment between the examined distributions. This is practical because all metadata distributions are 1-dimensional. This method serves as an effective tool in quantifying the degree of similarity between the metadata distributions in question, thereby facilitating a nuanced assessment of alignment accuracy within the datasets under review.

5.2.1 Bounding-box Area

Our preliminary metadata alignment effort pertains to the bounding-box area. Analyzing benchmark results from diverse datasets, it's discernible that models trained on synthetic datasets manifest a limitation when the real dataset includes close-ups not present in the synthetic counterparts. Specifically, these models often discern only segments of the object within its close-ups. An illustrative example is provided in figure 5.1.

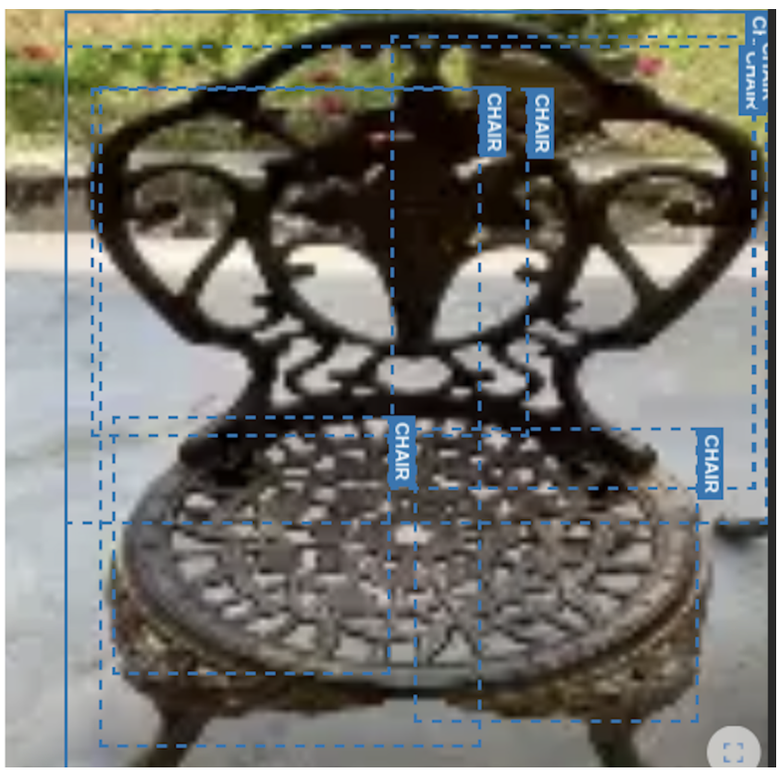


Figure 5.1: Predictions on chair's close-up

In addition, subsequent observations indicate a tendency for models trained on synthetic datasets, which contain a high concentration of small objects at their boundaries, to predict an excessive number of small bounding boxes at the edges of the test dataset. This phenomenon highlights the necessity to align the bounding box areas between the synthetic and real distributions more closely.

To address this, we propose an empirical assessment that incorporates three distinct versions of the synthetic dataset. These variations are crafted to facilitate a deeper exploration into the effects of bounding box distribution harmonization, thereby aiding in the optimization of model predictions when transitioning from synthetic to real-world data environments. This analytical

progression paves the way for a more grounded and practical approach to evaluating and enhancing the interoperability and alignment between synthetic and real datasets in machine learning applications.

1. **Baseline Dataset:** This straightforward dataset encompasses objects sampled at random, with an equivalently random camera placement. No explicit control is exercised over object size within image samples. As a result, numerous tiny object fragments are detectable on image boundaries due to artefacts resulting from random sampling, often occupying merely a handful of pixels but yet denoted as objects of interest in annotations. Notably, such tiny annotated objects on boundaries are almost absent in the real dataset.
2. **Filtered Dataset:** Mirroring the baseline dataset, albeit with a crucial distinction. Here, we remove tiny objects on the boundary captured by the camera with scant pixels. A concrete threshold (50 pixels) is set, and all objects whose area falls below this threshold are omitted from the images.
3. **Aligned Dataset:** This dataset is constructed utilizing our automated metadata alignment process during data generation, as expounded in the methodology chapter. We first ascertain the bounding-box area distribution using the real validation dataset, subsequently integrate this into the generation process, and meticulously align the synthetic bounding-box area with it.

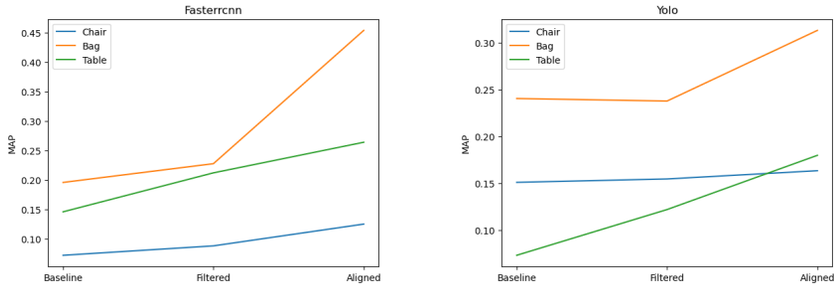
For these delineated scenarios, we instituted experiments across diverse problem constructs, namely: chair, table, and bag. Each problem construct is paired with a congruent real test dataset, consistent throughout the experimentation. The trio of dataset versions listed above are utilized for evaluative purposes.

Conclusive evaluative outcomes are tabled in 5.1. Values external to the brackets denote the MAP (Mean Average Precision) values for Faster R-CNN while those internal to brackets correspond to the YOLO model.

	Chair	Bag	Table
Baseline	0.072(0.1511)	0.1957(0.2404)	0.1458(0.1734)
Filtered	0.088(0.1547)	0.2276(0.2377)	0.2120(0.2221)
Aligned	**0.125**(**0.1635**)	**0.4540**(**0.3130**)	**0.2427**(**0.2709**)

Table 5.1: MAP with aligning bbox area

Additionally, plot 5.2 also shows the benchmark results of the three versions of synthetic datasets with Faster R-CNN and YOLO and the behaviour of



(a) Fasterrcnn (b) Yolo

Figure 5.2: MAP from Bounding-box Area benchmarks

	Chair	Bag	Table
Baseline	0.357	0.4727	0.4183
Filtered	**0.0584**	0.1064	**0.1217**
Aligned	0.076	**0.0912**	0.1447

Table 5.2: Percentage of False Positives with small bbox area in predictions

	Chair	Bag	Table
Baseline	0.5182	0.5497	0.4970
Filtered	0.4805	0.4774	0.4153
Aligned	**0.1616**	**0.1088**	**0.1368**

Table 5.3: Bh Distance in Bounding-box Area

models before/after alignments are shown more intuitively. The evaluation of alignment is shown in table 5.3.

Interpretation The alignment metrics show that the distributions after filtering and aligning match better with the real bbox-area distribution. Further, we can also observe that there is a significant performance gap between the aligned and unaligned datasets, especially with the Faster R-CNN and Bag datasets. Additionally, the table 5.7 shows the percentage of small bounding boxes in the predictions. The small bounding-boxes are defined as the False Positives in predictions with a bounding box area smaller than a certain threshold. From those evaluation results, we may conclude that the bounding-box area is essential metadata that should be aligned between synthetic and real distribution. An intuition behind this phenomenon could be that if the synthetic datasets consist of too many small objects, the model becomes too sensitive to signals in the background, which increases the number of false positives in predictions.

5.2.2 Image blur

The second metadata we benchmark is the image blur. Our experimental endeavors revealed a simple observation: incorporating blurring steps within the pre-processing sequence consistently enhanced model efficacy, a phe-

nomenon witnessed even when models were exclusively trained on synthetic datasets. This discernment underscored the potential benefits of meticulously aligning image blur in the context of synthetic image generation.

With this motivation as our backdrop, we introduce three distinct datasets, tailored for rigorous benchmark evaluation:

1. **Baseline Dataset**: It is the baseline dataset generated without aligning blur or augmenting with certain pre-processing steps.
2. **Pre-processed Dataset**: It is the same as the baseline dataset, but with blurring steps in the preprocessing pipeline using albumentation, i.e. in each randomly loaded data batch, a certain fraction of samples would be blurred randomly.
3. **Aligned Dataset**: It is the dataset with aligning blur on the image level by the automated pipeline described in the methodology section. The blurring is achieved by blurring out the image using Gaussian filters with standard deviation and probability of blurring as free parameters and the values for aligning are chosen. These images are preserved in this dataset and no extra blurring is done in the pre-processing steps.

The results are shown in the tables:

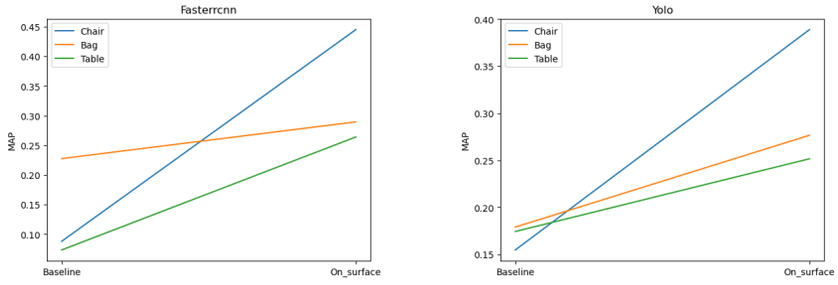
	Chair	Bag
Baseline	0.3783	0.4105
Filtered	**0.4169**	**0.4459**
Aligned	0.2609	0.4377

Table 5.4: MAP with aligning image blur

	Chair	Bag
Baseline	0.2148	0.5180
Filtered	0.2126	0.4510
Aligned	**0.2045**	**0.4187**

Table 5.5: BH-distance with image blur

Interpretation From table 5.5 we see that the alignment isn't working as significantly as aligning bounding-box area. It is expected because controlling blur by Gaussian filter is too coarse and fails to capture the actual semantic information underneath. Further, we can see that there is no correlation between the metric MAP and the alignment in blur. Note that adding blurring in the pre-processing step did increase the overall results because it increases the difficulty in training by varying the samples from batch to batch in the training pipeline. However, manually blurring a portion of the synthetic datasets didn't work well in both chair and bag datasets, as it did not increase the difficulty in training, since such blurred samples are fixed and the model could just memorize those cases. Therefore, we do not consider image-blur as meaningful metadata for aligning the distribution.



(a) Fasterrcnn (b) Yolo

Figure 5.3: MAP from Bounding-box Position benchmarks

Bounding-box Position and Object Orientation We now study the bounding-box positions and object orientations within datasets. Upon close inspection of various generated datasets and subsequent model evaluations, we identified significant discrepancies in object orientations and their respective bounding-box positions between synthetic and real datasets. Notably, this issue was markedly evident in datasets of chairs.

To illustrate, in real-world datasets, chairs predominantly appear grounded, which is a stark contrast to the synthetically generated datasets created through naive sampling methodologies. In the latter, chairs frequently manifested in the air or showcased unconventional orientations, such as an inverted stance. This discrepancy is depicted in figure 5.4.

To conquer this limitation in naive sampling strategy, efforts must be channelled towards ensuring both the orientation and position of objects accurately reflect the patterns found in real datasets. Consequently, we propose the evaluation of two specific datasets designed to benchmark the efficacy of methods aimed at achieving this alignment.

1. **Baseline Dataset**: the dataset with random sampling strategy for both object orientation and the location of the objects.
2. **On-surface Dataset**: the dataset with position sampling over a surface and object orientation only along the z-axis. Note that this is not generated with the automatic data pipeline, because we completely re-implement the sampling strategy and not only change a set of parameters.

The benchmarking results are shown in the table 5.6, where the one outside bracket is the MAP with Faster R-CNN, and the one inside is the MAP with YOLO. The Bhattacharyya distance only accepts only 1-dimensional distribution, and position distributions of objects are two-dimensional. However, since we did not change the sampling strategy along the horizontal axis, we only need to compute the Bhattacharyya distance on the vertical axis. Further,

Figure 5.4: A flipped and floating synthetic chair

	Chair	Bag	Table
Baseline	0.088(0.1547)	0.2276(0.2377)	0.1021(0.1743)
On-Surface	**0.4452**(**0.3889**)	**0.2895**(**0.2766**)	**0.3045**(**0.3721**)

Table 5.6: MAP with aligning bounding-box position

we use the middle point of the bounding box as the reference point for object coordination. The evaluation result is shown in the following table:

	Chair	Bag	Table
Baseline	0.2739	0.1334	0.2516
On Surface	**0.0775**	**0.1113**	**0.0932**

Table 5.7: BH-distance in vertical axis

Interpretation Plot 5.3 shows more intuitively the performance of the models(Faster R-CNN and YOLO) before and after the alignments. We can see that gain in performance significantly differs from dataset to dataset. How-

ever, the performance did increase overall after the alignments. An intuition behind this phenomenon is that the relative position between the camera and the object of interest matters. For instance, if the objects in the synthetic samples are mostly sampled above the camera, and the camera looks at the objects from a lower angle, it can be expected that the model trained on those samples would not be able to recognise those objects if the shots are taken above the objects from a higher angle, since the shape of the objects may dramatically different from a huge change in relative positions between camera and objects.

5.2.3 Evaluation of Other Basic Metadatas

Beyond the aforementioned metadata, a plethora of others warrants discussion, including but not limited to 'image sharpness', 'image entropy', 'bbox sharpness', 'bbox blur', and 'bbox entropy'. Nevertheless, the alignment of these metadata does not appear to offer tangible improvements when optimizing the Mean Average Precision (MAP) metric.

At an intuitive level, certain metadata, such as image entropy, offer limited insight into the distributions of our primary objects of interest. This is primarily due to its exclusion of object-specific or positional information. Moreover, the computation of image entropy does not consider the orders of pixels, thereby lacking the ability to encapsulate meaningful semantic content. Consequently, efforts expended on aligning such metadata would not help generate better synthetic image samples.

A similar sentiment extends to 'image contrast' and certain other metadata. On the other hand, another illustrative perspective on this matter can be garnered from specific histograms that detail metadata distribution across an assortment of datasets, each generated with distinct sampling strategies via our synthetic image creation pipeline. These image and bounding-box level metadata are visualized in histogram plots referenced as 5.5 and 5.6 respectively. The varying hues within these histograms correspond to the different dataset sampling strategies, annotations for which can be found at the histogram's top left corner. A consistent observation across these visual representations is the almost identical distribution of these metadata, regardless of the sampling strategy employed or the results garnered when benchmarking object detection algorithms on the respective synthetic datasets. However, the quality of those synthetic datasets varies, and the model performance varies a lot as well when training on those datasets. This again gives us hints that aligning those metadata explicitly in the training pipeline would not work out as desired.

Thus, the alignment of such metadata neither appears necessary nor seems to exert any substantial influence on model performance, primarily due to the dearth of meaningful information they encapsulate. As a concluding note,

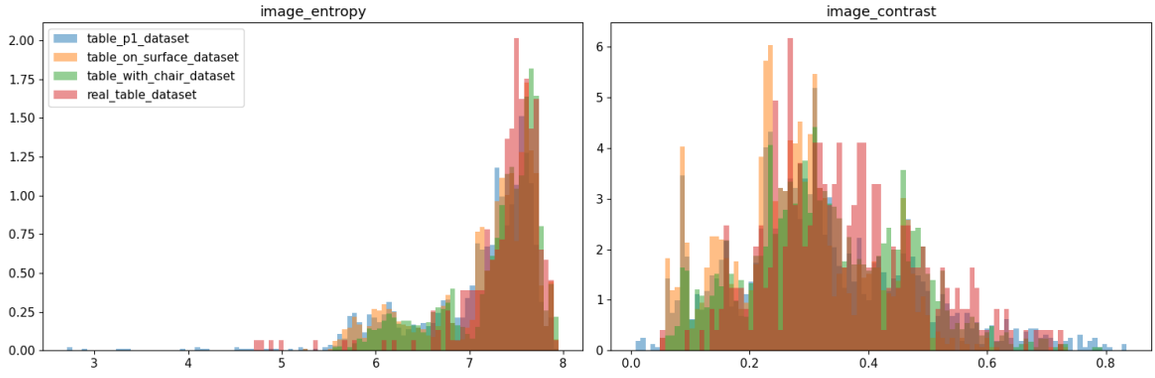


Figure 5.5: Image-level metadata distribution examples

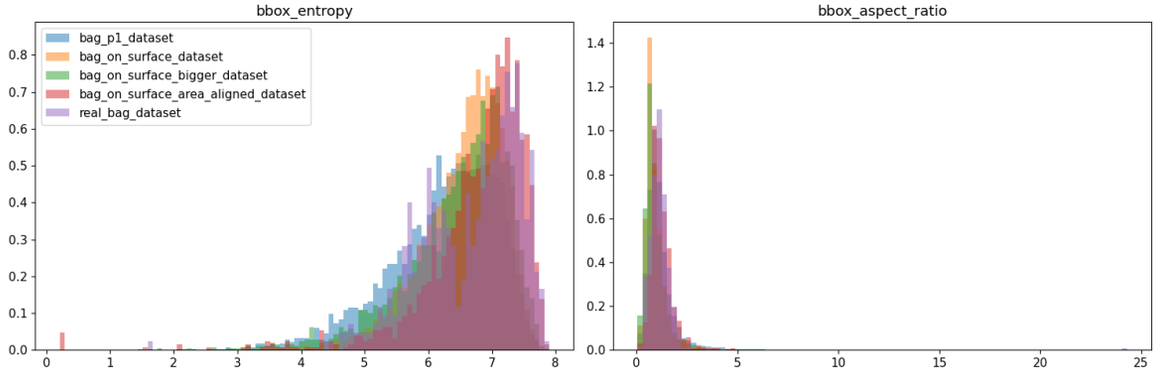


Figure 5.6: Bounding-box-level metadata distribution examples

we ascertain that these metadata may not be the most prudent choices for alignment endeavors.

5.3 Advanced Metadatas

Together with our Methodology section regarding advanced metadata 4.6.2, our exploration extended to the derivation of more sophisticated and insightful metadata from the synthetic datasets. These metadata capitalize on information harnessed directly from the rendering processes. One challenge, however, arises from the fact that real datasets lack explicit labels for these advanced metadata. To bridge this gap, as delineated in the Methodology, we used a feature-based approach, detailed in section 4.6.2. In line with this approach, our initial efforts were dedicated to benchmarking potential feature extractors. This was imperative to discern and subsequently adopt extractors that were performant enough to encapsulate the most meaningful information.

5.3.1 Benchmarking feature extractors

We would benchmark the feature extractors by fitting the features of the samples to the available metadata from the synthetic datasets and doing cross-validation for computing the evaluation metric. If the metadata are numeric values, the R2 score is used for evaluation, otherwise, we use the accuracy. As described in the section 4.7, we benchmarked both feature extractors for general purposes and object detections. In particular, we benchmark the Clip, Data2Vec and Dino feature extractors as examples of feature extractors with general purpose, as they are widely used in the industry. We benchmark the feature extractors for Faster R-CNN and YOLO, as they are representative of feature extractors from deep object detection models. We would also benchmark both pre-trained and trained feature extractors, as described in 4.7. The results of benchmarking are presented in table 5.9 and 5.8. The plot is shown in 5.7.

Interpretation We can make the following observations based on the outcomes:

1. The pre-trained feature extractors outperform the trained feature extractors on all metadata predictions. Therefore, we conclude that their features contain more information than the trained ones. An explanation for that is the trained feature extractors tend to view those attributes as invariant. In fact, in datasets and annotations, all those objects of interest are treated the same, and the model is trained to find out all such objects, regardless of their shape, depth, occlusion, and background similarity. Therefore, the trained feature extractors would map those objects close to each other in the feature space, which leads to poor predictions in those metadata values.
2. The pre-trained object detection feature extractors, Faster R-CNN and YOLO, outperform the general purposed feature extractors Clip, Data2Vec and Dino on predicting shape and occlusion, however fall short on depth and background similarity. An explanation for that is feature extractors specialized for object detection are good at extracting objects from the background. Therefore, it contains more information regarding the object rather than the background around it.

To conclude, for labelling the real datasets with advanced metadata, we use pre-trained feature extractors from object detection models for fitting shape and occlusion on synthetic datasets and for inferencing the metadata on real datasets. We use general-purpose feature extractors for fitting depth and background similarity on synthetic datasets and inferencing on real datasets. The trained feature extractors will not be used.

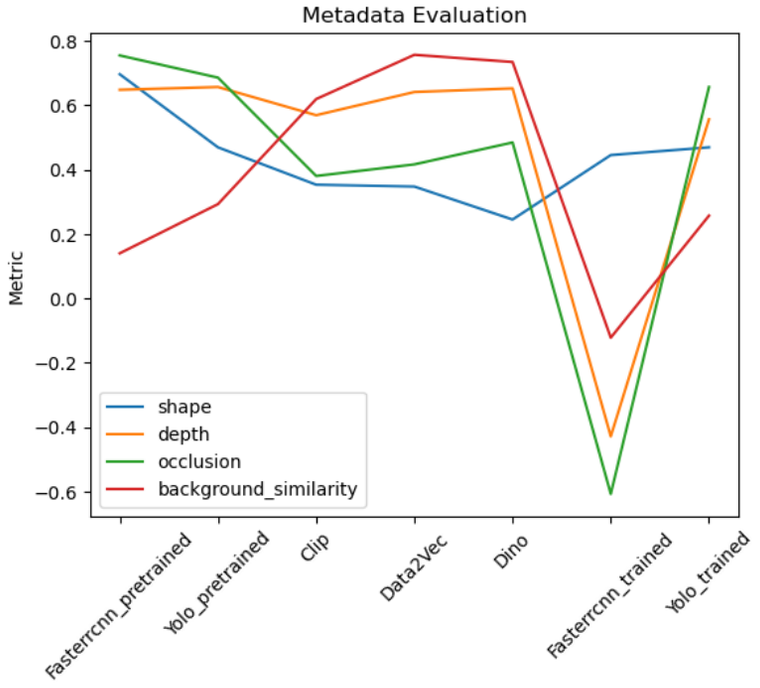


Figure 5.7: Benchmark Results of Feature Extractors with metadata prediction

	Shape	Depth	Occlusion	Background Similarity
$Fasterrcnn_{pretrained}$	0.6962	0.6479	**0.7547**	0.1398
$Fasterrcnn_{trained}$	0.4450	-0.4291	-0.6080	-0.1225
$Yolo_{pretrained}$	0.4691	**0.6566**	0.6857	0.2926
$Yolo_{trained}$	0.4740	0.5562	0.6566	0.2567
Clip	0.3529	0.5689	0.3801	0.6191
Data2Vec	0.3472	0.6411	0.4161	**0.7567**
Dino	0.2447	**0.6522**	0.4842	0.7343

Table 5.8: Evaluation of Feature Extractors on chair datsets

	Shape	Depth	Occlusion	Background Similarity
$Fasterrcnn_{pretrained}$	**0.8103**	0.2164	0.6640	-0.0227
$Fasterrcnn_{trained}$	0.8053	-6.5910	-3.8544	-9.8300
$Yolo_{pretrained}$	0.7235	0.1110	**0.6657**	0.1432
$Yolo_{trained}$	0.6355	0.2696	0.5735	-0.1618
$Clip$	0.4454	0.2456	0.3326	0.4418
$Data2Vec$	0.5012	0.2894	0.3785	**0.6399**
$Dino$	0.6112	**0.3813**	0.4791	0.4600

Table 5.9: Evaluation of Feature Extractors on bag datsets

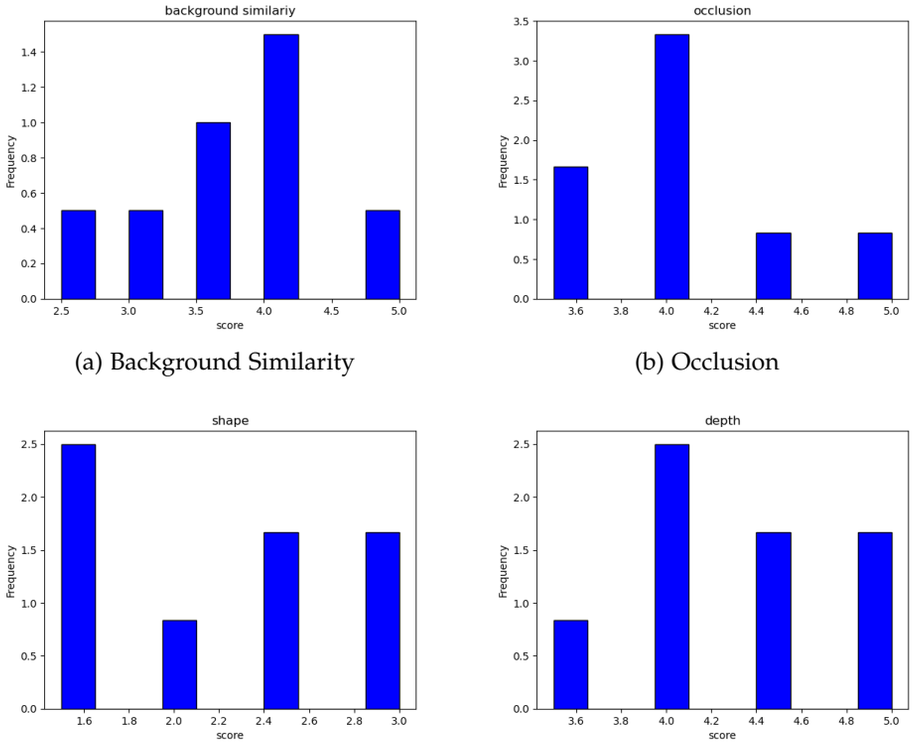
Accurately assessing the quality of predicted labels on real-world datasets presents a significant challenge, primarily due to the absence of ground-truth labels for direct comparison. Given this limitation, human evaluation emerges as a robust alternative to gauge the label quality.

For this study, we collected feedback from eight individual evaluators, exposing them to real samples adorned with the predicted labels. These labels were organized and presented based on their assigned values. The setup is simple: We use the widget in jupyter notebooks to visualize all real image samples, each annotated with the predicted metadata values. The evaluator goes through all real samples and evaluates whether the predicted values are accurate or not. Evaluators were then tasked with rating the accuracy of the labelling on a scale ranging from 0 to 5, where a score of 0 signifies poor labelling, while a score of 5 denotes exceptional accuracy. Since our real datasets are small, the process didn't take long. The aggregated results of this evaluation can be found in Figure 5.8.

Upon inspecting the histograms corresponding to the human evaluations, several observations come to the forefront. The predicted labels relating to background similarity, occlusion, and depth garnered favourable scores, indicating a satisfactory alignment with human expectations. However, a deviation in this trend is evident when evaluating the labels for shapes, which received notably lower scores.

A pivotal factor contributing to this discrepancy arises from the inherent challenges associated with achieving a one-to-one mapping between object shapes in synthetic datasets and their real-world counterparts. The synthetic objects utilized for this study were sourced from online 3D repositories. However, these repositories do not assure an exhaustive representation of all object shapes present in real datasets. Given this lack of comprehensive overlap, models trained on synthetic datasets exhibit limitations when predicting shapes in real datasets.

To address this challenge, a refined approach to object shape alignment is



(a) Background Similarity

(b) Occlusion

(c) Shape

(d) Depth

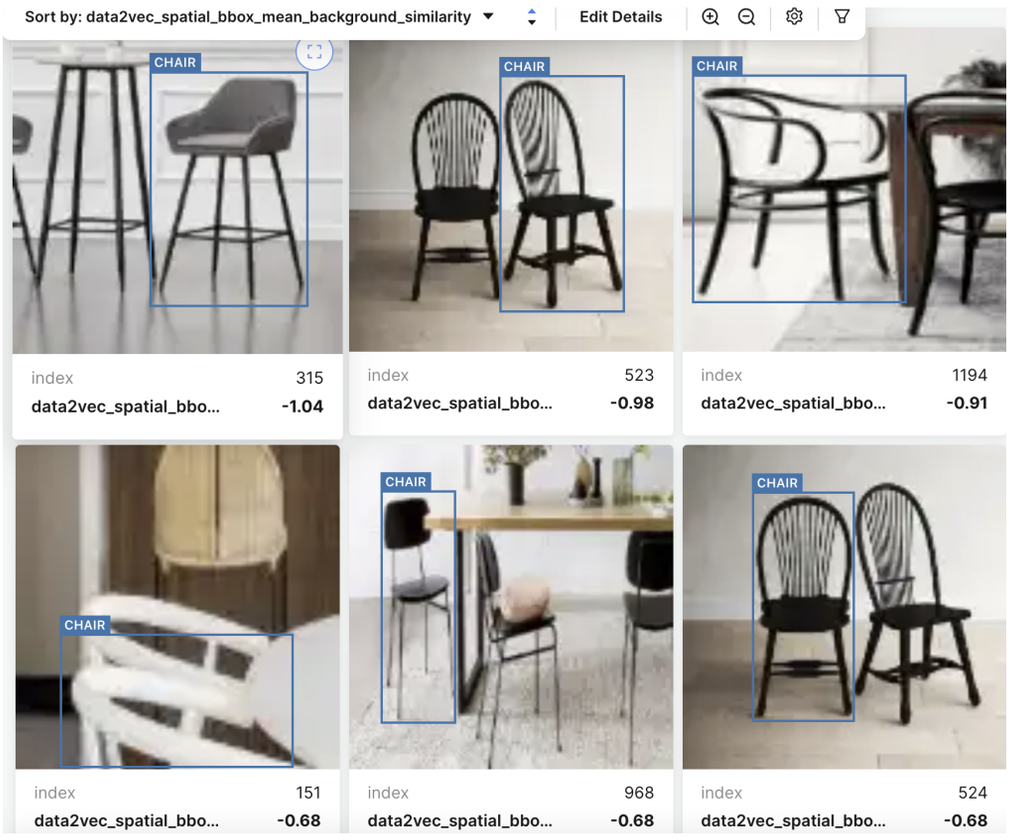
Figure 5.8: Human Evaluation for metadata prediction

essential. Detailed strategies for this alignment will be elucidated in the subsequent "feature alignment" section. For other metadata attributes, our proposed metadata alignment technique, as outlined in the Methodology section, remains pertinent.

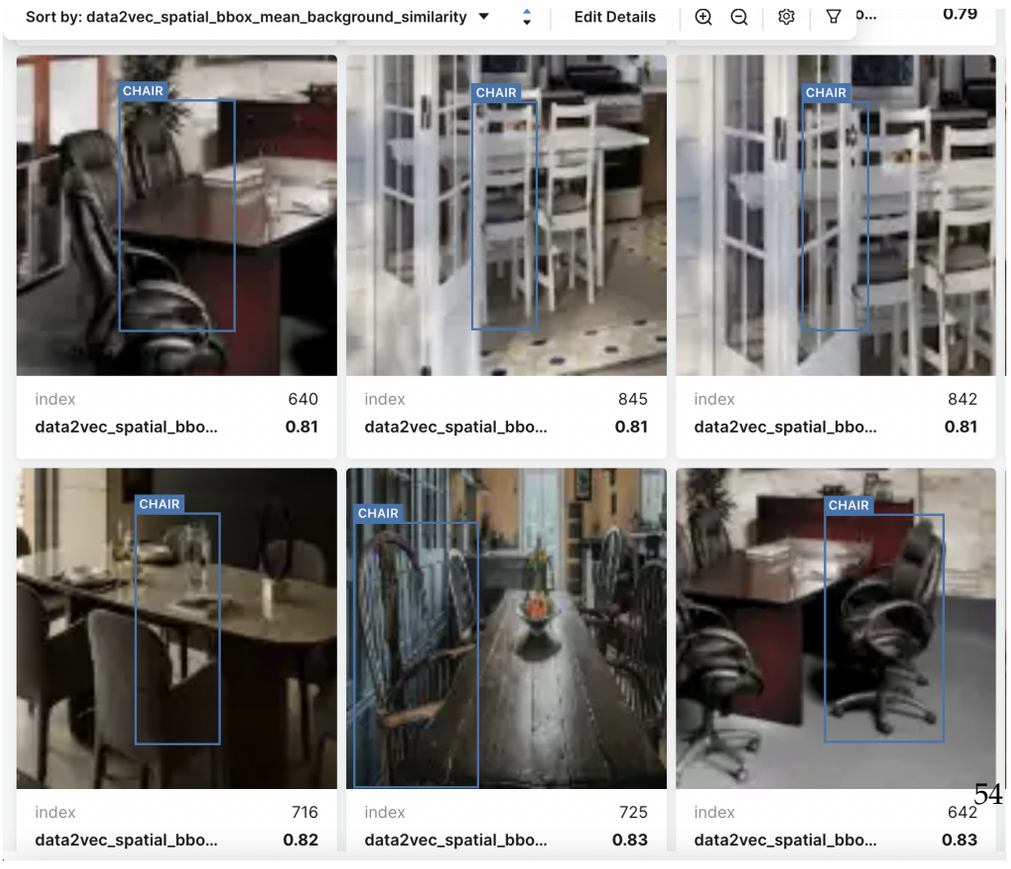
5.3.2 Background Similarity

As elaborated in the preceding chapter, the metric used to determine background similarity is derived from computing the cosine similarities between pixels located on the object boundaries and the surrounding background pixels. In this context, a lower similarity value implies a pronounced dissimilarity with the background, while a higher value suggests a closer resemblance.

Through rigorous benchmarking, the Data2Vec feature extractor emerged as the preferred choice for gauging background similarity in real-world datasets. Illustrative examples of these inferred background similarities can be viewed in Figure 5.9. A cursory inspection of the results underscores the high accuracy of the predicted labels. With the predicted labels we can now benchmark the background similarities. We propose the following three



(a) Low Similarity Score



(b) High Similarity Score

Figure 5.9: Predicted Similarity Scores across samples

versions of datasets:

1. **White-background**: The objects are all sampled on a white background.
2. **Baseline**: The objects are sampled using the same strategy but with selective backgrounds additionally sampled.
3. **Styled**: The sampling strategies are the same as baseline, but a style transfer is provided so that the objects are more similar to the backgrounds. The style transfer is done by the neural style transfer algorithm [61], with real samples as a reference for doing style transfer. The style transfer is done for the whole image.

	Chair	Table
White	0.0082(0.1011)	0.0260(0.0912)
Baseline	0.4211(**0.4006**)	**0.2640**(**0.2883**)
Styled	**0.4512**(0.3969)	0.2572(0.2544)

Table 5.10: MAP of chair and bag Datasets

	Chair	Table
White	0.7492	0.7246
Baseline	0.1557	0.2508
Styled	**0.1026**	**0.2236**

Table 5.11: Bh Distance in Background Similarity

The results are shown in the tables 5.10 and 5.11. In our assessment of object detection algorithms, the Mean Average Precision (MAP) values were gathered for both YOLO and Faster R-CNN. Observations point towards a notable discrepancy when considering datasets with white backgrounds. Specifically, these datasets seem to diverge significantly from the real distribution, leading to subpar MAP scores for both detection algorithms.

Interpretation On comparing the baseline datasets with styled ones, no evident correlation between background alignment and performance emerges. In particular, YOLO's performance downgrades on the styled datasets, even though their background similarities appear more aligned with the real distribution. This outcome is further highlighted in figure 5.10, showcasing a sample from the styled dataset. Despite feeding authentic image samples to the neural style transfer mechanism, the resultant image projects an artistic flavour rather than a realistic depiction. However, the background similarities align more closely with expectations, because there is an artistic harmony between the objects with the background behind them after the style transfer, although a human would view it otherwise: he would say that the image is

less realistic after the style transfer. This example shows that background similarity can not be used standalone as an effective metadata. However, the examples with white-background datasets show us that the background similarities are very unaligned with the real images. In those cases, the model would just classify any signal changes in the images as an object of interest, which leads to extremely poor results.

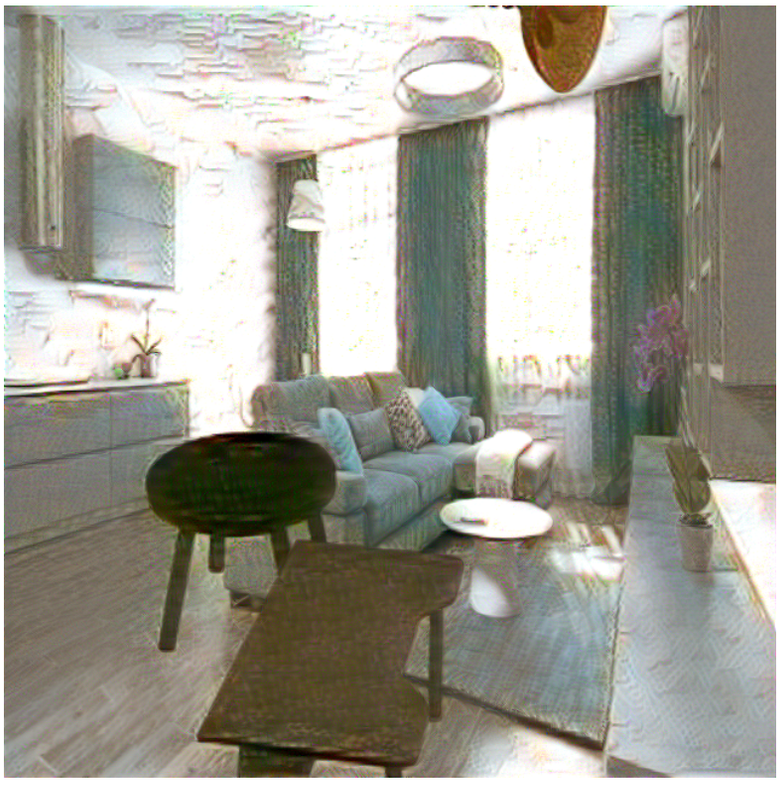


Figure 5.10: An example of image samples in the styled datasets

5.3.3 Depth

In the progression of our research, an interesting observation about the depth was identified. We noticed that close-up views of objects were frequently misclassified as multiple entities by the object detection algorithms, as shown in figure 5.1. To address this anomaly, we incorporated a subset of specifically generated close-ups into our datasets and subsequently evaluated the models.

For this particular evaluation, we diverged from the conventional MAP metric. Instead, we define a new metric which counts the number of overlapping predicted bounding-boxes appearing on the same close-up within the test datasets. To establish a comprehensive understanding, we introduced a comparative evaluation with the FCOS model [51], a model renowned for its

wide practical application, as well as Faster R-CNN and YOLO. The compiled results are encapsulated in the table below:

	Baseline	With close-ups augmented
Faster R-CNN	307	**75**
FCOS	812	**156**
YOLO	34	**28**

Table 5.12: Number of overlapping predictions on close-up test samples

Interpretation An evident decline in the number of overlapping predictions on close-up test samples is observed upon augmenting the datasets with close-ups. Notwithstanding, it is imperative to note the considerable variance in outcomes based on the choice of object detection algorithms. The FCOS model, for instance, exhibited a high number of bounding box predictions in close-up scenarios, emphasizing the efficacy of dataset augmentation. In contrast, the YOLO model showcased minimal discrepancies before and after the augmentation, suggesting that the impact of close-ups is relatively muted in its application.

5.3.4 Other Advanced Metadata

Beyond the commonly utilized metadata, there exist more intricate ones, such as the object's pixel area and its shape. While we approached the shape using a feature-based methodology, which will be elaborated upon in the subsequent section, the results for other metadata types were not promising. Taking occlusion as an example: even though occluded objects generally register the highest prediction errors compared to other test samples, aligning the occlusions in the datasets - whether by augmenting them or through a semi-automated approach - did not notably improve the model's performance on our benchmarks, which is counter-intuitive. A possible explanation might be the intrinsic difficulty in deciphering partially hidden samples. Augmenting datasets to accommodate these challenges may not provide a significant advantage, as the model would simply memorize those difficult examples in the training dataset since the occlusions do not vary in pre-processing steps. As such, we deduced that these metadata types are not instrumental for aligning the underlying data distribution. However, for object shape, we would have another study in section 5.4.2, because we take a completely different approach to reducing gaps between the distribution of shapes in synthetic and real datasets. The reason for that is, as described above, the predicted shape values on real datasets are not good enough according to human evaluation.

5.4 Feature Distribution Alignment

Pre-trained feature structures, when applied to data samples, usually produce information-rich features. Owing to their comprehensive training on expansive datasets, these feature extractors encompass prior knowledge of image analysis, object detection, and related domains. The content within these features typically surpasses the richness of human-generated or computed metadata. As a result, aligning the feature distributions of data samples becomes imperative. This alignment could be much easier and more effective than directly aligning the image distribution due to its lower dimensions and favourable signal-to-noise ratio, as the pre-trained feature extractors filter out the irrelevant noises in embedding and preserve primarily necessary image semantics. Nevertheless, it poses its challenges, especially when compared to metadata alignment, because it still has higher dimensions, usually varying from 512 to 2048.

5.4.1 Overview

The underlying principle of feature alignment rests on approximating the feature distribution. Direct alignment is a complex task given the high-dimensional nature of features. An effective workaround involves utilizing heuristic measures like K nearest neighbours (KNN) [62] and kernel density estimation models [46]. Specifically, we train either the KNN or kernel density estimation model on real datasets. Leveraging this, we can evaluate synthetic samples against the model by calculating the probability it is drawn from the real feature distribution, deducing whether a given synthetic sample is proximal to the real distribution based on its likelihood of being drawn from the actual feature distribution. Consequently, this could be viewed as the closeness of each synthetic sample to real distribution and be assigned as a pseudo-quality score. With this piece of information, we can filter out images that are less likely drawn from the real data distribution, or augment the dataset with samples that are more likely drawn. This could be viewed as a post-processing step.

5.4.2 Feature-based Metadata Alignment

Shape

For feature-based metadata alignment, we leverage features to assign a pseudo-quality score using density estimation to each synthetic data sample, as described above. Afterwards, we can aggregate those pseudo-quality scores according to certain metadata, i.e. we group the samples according to their metadata values and take the mean of the score for each metadata value. The interesting case occurs when a certain metadata value has a much lower average score than other metadata values. A plausible deduction from such

an observation could be: that these metadata values may not be represented in the real distributions. Hence, discarding these metadata values during generation may achieve better alignment.

This methodology is especially relevant for categorical metadata values such as shape. It's crucial to note that the shapes of objects in synthetic datasets might not necessarily mirror those in real-world datasets. Traditional prediction mechanisms, which rely on fitting and predicting using synthetic datasets and feature structures, fall short in such instances, as we saw from the human evaluation plot 5.8. Although the trained classifier is very good at predicting shape on synthetic datasets, as shown in table 5.9 and 5.8, they fall short when the domain changes to the real dataset, as the classes of shapes changes. To overcome this limitation, a feature-based KDE model is employed to implement the approach described above in 5.4.2. For every distinct shape, the quality score of samples in the synthetic dataset corresponding to that shape is aggregated. Our prime focus is to identify specific metadata sets that consistently register poorer quality scores relative to others. In our cases, we remove the worst two shapes from the synthetic datasets based on the aggregated quality scores.

Interpretation In Table 5.13, we detail our findings before and after the exclusion of shapes that deviate substantially from the real feature distribution. For each dataset scrutinized, the two most aberrant shapes were eliminated. A comparative analysis reveals that the model's Mean Average Precision (MAP) escalates appreciably post-alignment for both Faster R-CNN and YOLO algorithms across all synthetic datasets. From this analysis, it is evident that object shapes are pivotal metadata attributes demanding alignment.

	Chair	Table	Bag
Baseline	0.4293(0.3675)	0.3168(0.3685)	0.4540(0.3130)
Aligned	**0.4626**(**0.4104**)	**0.3349**(**0.3883**)	**0.4579**(**0.3223**)

Table 5.13: MAP before and after aligning shapes with features

5.4.3 Interlude

Dealing with high-dimensional feature distributions directly poses certain challenges. In light of this, we have identified two key strategies to mitigate these complexities, filtering and augmentation, as described in 5.4. The filtering approach involves critically examining our synthetic data. It's about filtering out the noise and retaining only the most representative samples.

5.4.4 Filtering

The underlying principle of the filtering strategy is relatively direct. Employing a feature extractor, features are extracted from real data. Subsequently, models like the K-Nearest-Neighbour [62] or the Kernel Density Estimation [63] are fitted using these features. For each synthetic data sample, the probability of its derivation from the real feature distribution is computed. Once all samples have associated probabilities, a portion of them is discarded using the automated elbow method. It's essential to understand that this filtering technique isn't exclusively reserved for juxtaposing synthetic versus real datasets. Notably, it is versatile enough for scenarios where test datasets exhibit divergent distributions from their training counterparts. Consequently, this strategy's efficacy will also be assessed on established real datasets, including the COCO dataset [64].

Two distinct levels of filtering come into play here. On one hand, we have image-level filtering, wherein each bounding box's probability is amalgamated at the image level. Using the elbow method, the least representative portion of images in the training datasets is then filtered out. On the other hand, bounding box-level filtering focuses on the direct elimination or modification of specific bounding boxes. In practice, this means making the least representative bounding boxes invisible in the datasets by removing their annotations or rendering them in black.

Synthetic Datasets

In our analysis of synthetic datasets, we label our datasets as "Baseline", "Filtered", and "Black-boxed". The outcomes of our experiments are tabulated for easy reference. We initiated our investigation with data filtering, specifically focusing on the 'chair' and 'bag' categories. The performance metrics were subsequently ascertained on real test data, which remained unfiltered. The results are displayed in Table 5.14.

Interpretation Upon reviewing the results, we observe a discernible enhancement in dataset performance post-filtration. However, the black-boxing technique yielded mixed outcomes. A plausible reason for this variation might be the degradation of the semantic quality of the images upon black-boxing. Furthermore, the presence of overlapping objects complicates matters; black-boxing one object inadvertently impacts the adjacent ones.

Real Datasets

To evaluate our filtering method using the COCO datasets, it's imperative to discern the distribution differences between objects of interest in the test datasets as compared to the training sets. To achieve this, we modified the

	Chair	Bag
Baseline	0.4293	0.4540
Filtered	**0.4647**	**0.4601**
Black-boxed	0.4388	0.4490

Table 5.14: MAP before and after filtering and black-boxing

annotations, preserving only one class in the test dataset. Consequently, this makes the test object distribution notably distinct from the training object distributions.

For a comprehensive evaluation, we compared our approach against multiple baselines. The unaltered dataset, labeled the ”original dataset,” served as our primary point of reference. The first baseline termed the ”subsampled train dataset,” involved randomly removing 20% of the images from the training set. Next, the ”single class dataset” retained only the class present in the test set from the training annotations. The third baseline, the ”cleaned dataset,” randomly removed 20% of the training images, ensuring the elimination of images devoid of any class present in the test set. An illustration is shown in figure 5.11

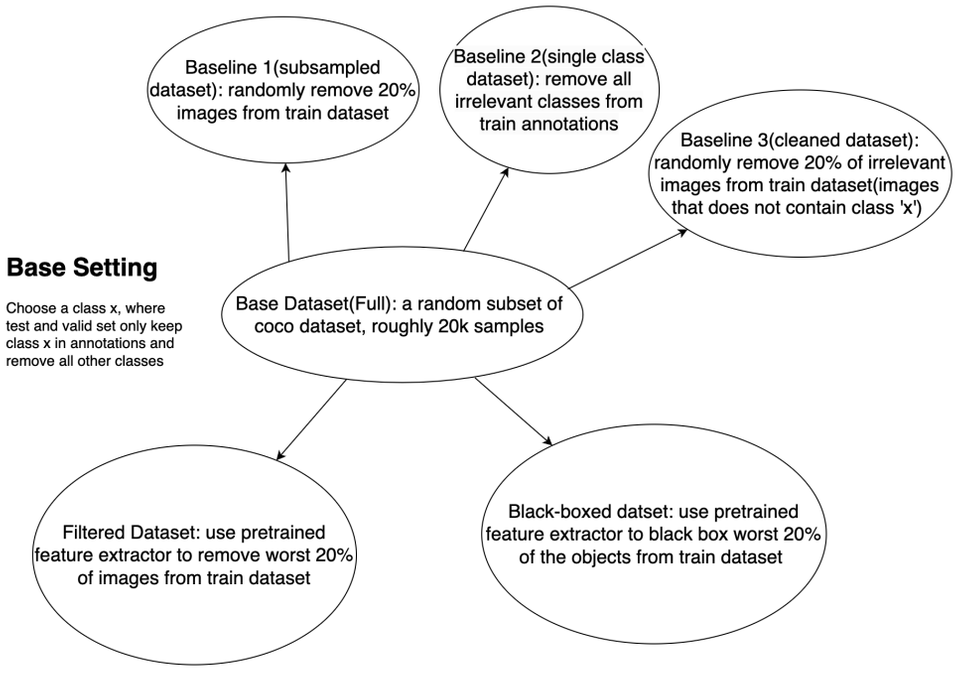


Figure 5.11: The baselines and proposed datasets for evaluating feature-based filtering approach on COCO dataset [64]

Contrasting these baselines, we proposed two filtered datasets. The first filters the least representative 20% of images, and the second black-boxes remove annotations corresponding to the least representative 20% of bounding boxes. We standardized the removal percentage at 20% to ensure equitable comparison. Post-filtration, we observed a reduction of approximately two thousand samples in comparison to the original datasets.

Interpretation The results of this benchmarking exercise are tabulated in Table 5.15. Two test set scenarios were considered: one preserving only 'cars' and the other retaining only 'people.' Our filtration methods not only surpassed the performance of the subsampled train dataset but also exhibited superior results compared to the manually curated datasets, and on occasions, even the full datasets. A possible rationale for this could be the inadvertent benefit of retaining certain challenging samples that share similarities with classes in the test dataset. For instance, in distinguishing between a car and a bus, it's beneficial to retain images of buses that resemble cars. Such nuanced distinctions, which manual filtration might overlook, are capably handled by our filtering method.

	Person	Car
$Full$	0.2599	0.1650
$Baseline_{singleclass}$	0.2694	0.1460
$Baseline_{subsampled}$	0.1742	0.0843
$Baseline_{cleaned}$	0.2627	0.1547
$Filtered$	**0.2846**	**0.1665**
$Black-boxed$	0.2654	0.1604

Table 5.15: MAP with Car and Person in COCO

5.4.5 Augmentation

Augmenting synthetic datasets through insights from true feature distribution augmentation is notably more challenging than simple filtering processes. It proves tough to generate samples that are more likely to be sampled real distributions using non-parametric distribution fitting, instead of just eliminating outliers. Efforts have been made and the process took a long time since we need to iteratively generate new samples based on evaluation of the previous version of datasets. Unfortunately, these augmentation techniques have not demonstrated a notable performance improvement when tested on our proposed datasets.

Although our current investigations have not achieved the desired outcomes, we believe potential advancements may still be uncovered in the domain

of dataset augmentation. Based on our current data, we advise a cautious approach to utilizing these techniques.

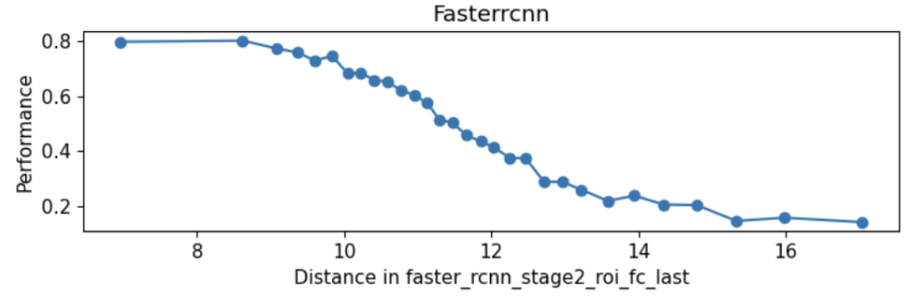
5.4.6 Sample-wise Performance and Feature Distribution Distance

During our experiments and benchmarking, an intriguing observation emerged: there appears to be a relationship between prediction quality for data samples and their proximity to the training dataset's distribution. Specifically, when a test sample aligns closely with the synthetic distribution used for training, the model tends to produce more accurate predictions for that sample. Conversely, if a test sample deviates significantly from the training distribution, the prediction quality tends to decline.

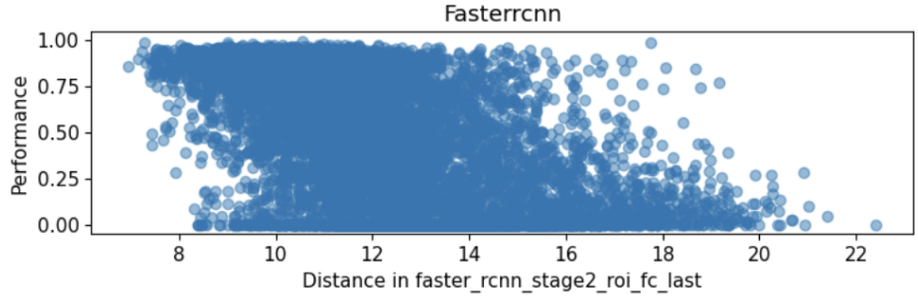
This observation has encouraged us to explore feature-based approaches. We have undertaken a systematic evaluation to quantify the correlation between individual data sample predictions and their distance from the training distribution. To measure this distance, we employed the K-Nearest Neighbors (K-NN) distance as our distance metric for quantifying the distance from the real sample to the synthetic clusters. Concurrently, we utilized the Intersection over Union (IoU) of bounding boxes as the sample-wise prediction evaluation metric. Note that this evaluation metric is limited, because it only focuses on the True Positives and False Negatives, as it is only calculated around the true objects of interest. However, we didn't find a way to also consider False Positives. Our analysis, visualized through bin plots and scatter plots, is detailed in Figures 5.12 and 5.13.

The visual data presented in the figures provides insight into the relationship between sample-wise performance and their respective distances to real samples. Specifically, for the chair dataset, a strong correlation is evident. On the other hand, while the correlation is less pronounced for the bag dataset, it remains discernible.

Upon closer examination, particularly when considering the scatter plot, several nuances become apparent. Despite the bin-wise plots indicating a clear correlation, the scatter plot reveals significant variability within each bin. This suggests high noise levels, overshadowing the underlying trend. The signal-to-noise ratio is notably low, which poses challenges in drawing definitive conclusions for individual samples based solely on their distances in the feature space. This observation underscores the complexities inherent in leveraging distribution distances in the feature space as a predictor of sample-wise performance.



(a) Bin Plot



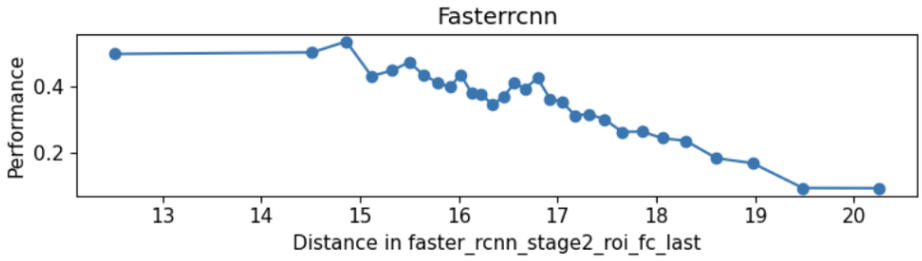
(b) Satter Plot

Figure 5.12: Chair dataset sample-wise performance vs distance to real distribution

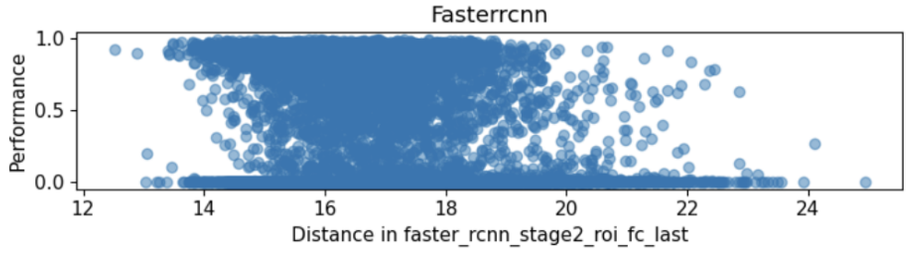
5.5 Evaluating the quality of synthetic datasets without training

In the course of our study, after benchmarking both the metadata and the features vital for gauging distribution distances between synthetic and real datasets, we recognized a potential approach for synthetic data quality evaluation. Specifically, by inspecting the alignment of pertinent metadata in the datasets under consideration, we can deduce information about dataset quality. Additionally, by leveraging the feature distribution distances between synthetic and real datasets, we can compute a composite distance for each dataset, which is computed as the mean of the metadata distribution distances among the metadata we benchmarked as significant and sample-wise KNN distances in feature space, extracted from pre-trained feature extractors, which is referred as distance in the following paragraphs.

The rationale behind this distance metric is its potential to provide insights into the quality of individual datasets without the necessity of model training. To validate this, we plot the distance against the Mean Average Precision (MAP). Here, the MAP of a synthetic dataset denotes the MAP attained on the real dataset when a model trained on synthetic data is tested.



(a) Bin Plot



(b) Satter Plot

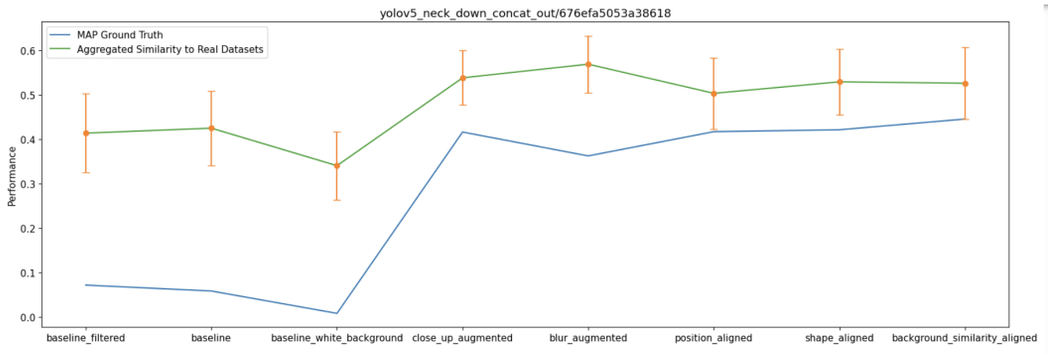
Figure 5.13: Bag dataset sample-wise performance vs distance to real distribution

However, our results present a nuanced picture. While there exists a discernible correlation between the distance and the algorithmic performance on those datasets, the strength of this correlation is variable. Specifically, for datasets where the model performance (i.e., MAP) exhibits stark differences, a conspicuous gap in the distances can be detected. Conversely, for datasets with closely aligned MAP values, the distance metric offers little to no discernment, and the correlation can even appear locally the other way around.

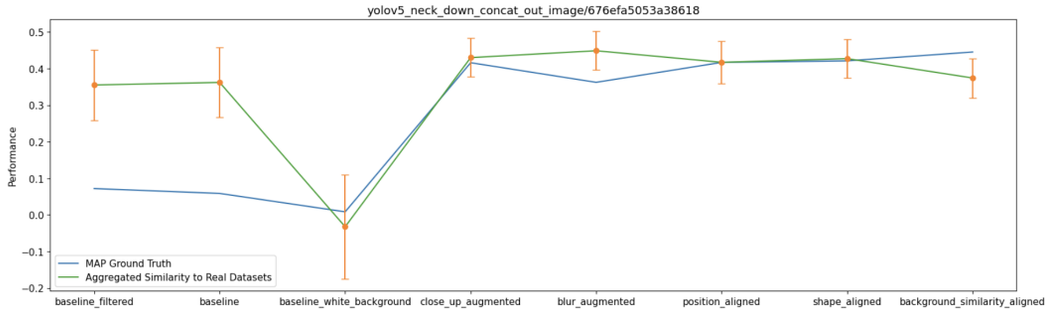
Refer to Figure 5.14 for a graphical representation of these findings. In this plot, the blue curve represents the MAP, and the green curve represents the distance. Because the distance is computed by taking the mean, we can also compute its variance for each dataset, which is denoted with the orange bar for each point on the green curve. A noteworthy observation is the considerable variance in these estimates, adding another layer of complexity to the interpretation of results.

5.5.1 Limitations of feature-based approaches

In our exploration of feature-based methods, we encountered a limitation tied to the dependency of feature extractors on the object class existing in their pre-training datasets. This implies that if a specific object class is absent



(a) Chair



(b) Bag

Figure 5.14: Dataset Evaluation on synthetic bag and chair datasets with predictions and ground truths

	Shape	Depth	Occlusion	Background Similarity
$Fasterrcnn_{pretrained}$	0.1368	0.3243	0.2292	0.1178
$Yolo_{pretrained}$	0.1031	0.3319	0.2854	0.1552

Table 5.16: Evaluation of pre-trained feature extractors on pistol dataset

during the pre-training phase, such as within datasets like coco or ImageNet, the feature extractor will likely struggle to garner meaningful insights related to that class. A concrete example of this limitation arose with pistols. Notably, our pre-trained feature extractors had no prior exposure to pistols within the coco dataset, leading to ineffectiveness when the object of interest spanned both real and synthetic data distributions. We evaluated feature extractors on pistol datasets, and as illustrated in the table 5.16, the outcomes were far from satisfactory. This underperformance was anticipated given the feature extractor's unfamiliarity with the object class in question.

	Chair	Table	Bag
Synthetic	**0.4647**(0.4216)	0.3349(0.3883)	**0.4601**(0.3928)
Real	0.4528(**0.4702**)	**0.3664**(**0.4225**)	0.4543(**0.4416**)

Table 5.17: Synthetic Dataset Performance vs Real Dataset Performance

5.6 Final results

Through extensive experimentation and benchmarking, we arrived at several key conclusions. Vital metadata, such as bounding box dimensions, positions, orientations, background similarities, and the objects' shape and occlusion, provide critical insights about the underlying data distribution. Hence, ensuring the alignment of these metadata distributions is crucial when fabricating synthetic datasets. Additionally, feature-based methodologies have exhibited notable performance enhancements. Specifically, feature-based distances disclose pivotal details about distributional disparities between synthetic and real datasets. Techniques like filtering and augmentation, rooted in feature distributions, can effectively adjust the synthetic feature distribution to align more closely with the real distribution. We also noted that applying the filtering technique to real datasets yielded appreciable performance boosts, as evidenced in the coco dataset evaluations.

When we combine these metrics, insights into dataset quality can be gleaned without necessitating algorithmic training. Our concluding results juxtapose the performance of our synthetic datasets against that of comprehensive real datasets. Here, "full real datasets" denotes models that were trained not merely on a subset but on a predominant fraction of the 500 real samples. Table 5.17 provides a detailed breakdown of these findings. Parenthetical values denote the MAP of YOLO, while non-parenthetical ones pertain to Faster R-CNN,. Encouragingly, our refined approach delivered results that rival those obtained on genuine datasets, with the Faster R-CNN model being especially commendable. It is worth noting, however, that the YOLO model exhibited a minor performance disparity compared to its real dataset-trained counterpart. In summation, models trained on both synthetic and real datasets offer largely analogous performances.

Chapter 6

Conclusion

6.1 Summary

In this study, we have addressed several critical aspects of synthetic image generation and dataset augmentation, elucidating potential paths to narrow the disparity between synthetic and real data distributions. Here, we summarize the primary findings of our research:

1. **Guidelines for Synthetic Image Generation:** We have devised guidelines for generating synthetic images when only a limited real dataset is available. A typical procedure encompasses the creation of a baseline dataset, which is utilized to fit and predict metadata on the real dataset. Subsequently, the inferred metadata aids in generating a second synthetic dataset characterized by aligned metadata distribution and curated object shapes. Applying this procedure across various synthetic datasets yielded results comparable to real datasets, specifically when evaluated using the Faster R-CNN and YOLO algorithms.
2. **Effectiveness of Metadata in Bridging Synthetic and Real Distributions:** Our study evaluated the efficacy of metadata in mitigating the gaps between synthetic and real data distributions. Notably, several metadata aspects such as bounding-box area, object position and orientation, background similarity, and depth and shape attributes emerged as potent factors in this regard, across different object detection algorithms and datasets.
3. **Feature-Based Filtering:** Our investigations have revealed that filtering based on sampling likelihood within the feature space is an effective strategy across diverse datasets, encompassing both synthetic and real samples. This is particularly valid when there is a distribution shift between the training and testing datasets. This technique can serve as a standard method for post-processing existing datasets or as a tool to

evaluate the quality of individual samples.

4. **Correlation between Sample Distances and Bounding-Box IoU:** A notable correlation has been observed between the distances of synthetic image samples to the synthetic feature distribution and the sample-wise bounding-box IoU in model predictions. Specifically, a real sample that closely aligns with the synthetic distribution in the training datasets is more likely to be accurately predicted by the model.
5. **Development of Synthetic Image Generation Pipeline:** We also established a sophisticated synthetic image generation pipeline, which leverages a semi-automated sampling strategy to effectively align the metadata distribution.
6. **Quantitative Analysis and Benchmarking Tools:** We developed methods for the quantitative analysis of datasets and the ensuing model outcomes. This includes the formulation of comprehensive benchmarking, metadata inferencing, and post-processing pipelines. These developed systems stand as instrumental tools in evaluating and enhancing the overall quality and reliability of synthetic datasets.

6.2 Limitations

While this study illuminates several significant avenues for advancing synthetic data generation and utilization, it is not without its limitations, as enumerated below:

1. **Interpretability of Feature-Based Approaches:** One of the prominent limitations observed was the lack of interpretability ingrained in the feature-based approaches, which can potentially restrict the comprehensive understanding and further refinement of the methodologies employed.
2. **Suboptimal Outcomes with Synthetic Dataset Augmentation:** The study witnessed a lack of success in augmenting synthetic datasets based on feature distributions. This endeavor did not yield the anticipated improvements, signalling a need for further research and optimization in this area.
3. **Constraints in Sample Generation:** The sample generation in this study was confined to the use of Blender and BlenderProc. The exploration did not extend to other potent engine-based renderers and replicators, such as the Nvidia Omniverse Replicators, which might offer different perspectives and possibly enhanced outcomes.
4. **Absence of Feature Distribution Alignment in the Generation Pipeline:** A significant gap in the current study is the non-incorporation of feature

distribution alignment within the synthetic image generation pipeline. This omission might potentially hinder the attainment of higher accuracy and alignment with real datasets.

5. **Lack of Effective Evaluation Strategy:** Despite the considerable progress achieved, the study did not manage to devise an effective strategy for evaluating the quality of datasets without the necessity to train algorithms on them. This indicates a crucial area where future research can focus on fostering a more nuanced and efficient evaluation framework. Nonetheless, it is prudent to acknowledge a pivotal assumption underlying our research— the inherent scepticism towards the existence of a universally effective strategy. We posit that the discovery of such a strategy would essentially enable the circumvention of the complexities associated with deep learning.

These limitations represent avenues for potential future work, where further innovation and refinement can lead to the development of more robust and reliable techniques in the synthetic data generation and analysis domain.

6.3 Outlooks

In the nascent field of synthetic image generation centered around a data-centric perspective, we stand on the threshold of numerous discoveries waiting to be unearthed. The potential for breaking new ground is substantial, especially when we consider the viable research avenues presented by the limitations detailed in the preceding sections. Alongside these, there lies the possibility of pioneering entirely novel approaches to address the current challenges:

1. **Venturing into Other Computer Vision Tasks:** The present study exclusively hones in on object detection algorithms. There is a palpable need to broaden this scope and extend similar investigative paradigms to other realms within computer vision, encompassing anomaly detection, image classification, and semantic segmentation. Expanding the research dimensions in this manner could unveil nuanced strategies and solutions, pushing the boundaries of our current understanding and applications.

2. **Incorporating Deep Generative Models into the Synthetic Image Pipeline:** At present, deep generative models are restricted in their ability to offer annotations and precise configurability. However, their proficiency in discerning and capturing implicit distributions stands as an untapped resource. It is proposed that future studies explore the integration of these models into the synthetic image pipeline, utilizing their strengths in grasping complex distributions to enhance alignment

strategies, potentially bringing about a paradigm shift in synthetic image generation methodologies.

3. **Exploring the Integration of Synthetic and Real Datasets:** This research maintained a purist approach, concentrating solely on training models using synthetic datasets. A logical and potentially rewarding progression would be to delve into studies examining the complementary roles synthetic and real datasets can play when integrated within a training framework. Such initiatives could illuminate critical facets that govern the synergistic interaction between synthetic and real samples, potentially laying the groundwork for more robust and comprehensive training datasets.

As we navigate forward, we must remain adaptable and open to uncovering innovative pathways. We contend that pursuing the avenues outlined above may not only circumvent current limitations but potentially spawn a new era of research, rich in depth and breadth, in the dynamic landscape of synthetic image generation and machine learning.

Bibliography

[1] M. Hassaballah and A. I. Awad, *Deep learning in computer vision: principles and applications*. CRC Press, 2020.

[2] I. Aganj, M. G. Harisinghani, R. Weissleder, and B. Fischl, "Unsupervised medical image segmentation based on the local center of mass", *Scientific reports*, vol. 8, no. 1, p. 13 012, 2018.

[3] M. Alloghani, D. Al-Jumeily, J. Mustafina, A. Hussain, and A. J. Aljaaf, "A systematic review on supervised and unsupervised machine learning algorithms for data science", *Supervised and unsupervised learning for data science*, pp. 3–21, 2020.

[4] A. B. Abdusalomov, R. Nasimov, N. Nasimova, B. Muminov, and T. K. Whangbo, "Evaluating synthetic medical images using artificial intelligence with the gan algorithm", *Sensors*, vol. 23, no. 7, p. 3440, 2023.

[5] S. Borkman, A. Crespi, S. Dhakad, S. Ganguly, J. Hogins, Y.-C. Jhang, M. Kamalzadeh, B. Li, S. Leal, P. Parisi, *et al.*, "Unity perception: Generate synthetic data for computer vision", *arXiv preprint arXiv:2107.04259*, 2021.

[6] R. J. Chen, M. Y. Lu, T. Y. Chen, D. F. Williamson, and F. Mahmood, "Synthetic data in machine learning for medicine and healthcare", *Nature Biomedical Engineering*, vol. 5, no. 6, pp. 493–497, 2021.

[7] H. Hattori, N. Lee, V. N. Boddeti, F. Beainy, K. M. Kitani, and T. Kanade, "Synthesizing a scene-specific pedestrian detector and pose estimator for static video surveillance: Can we learn pedestrian detectors and pose estimators without real data?", *International Journal of Computer Vision*, vol. 126, pp. 1027–1044, 2018.

[8] S. Tripathi, S. Chandra, A. Agrawal, A. Tyagi, J. M. Rehg, and V. Chari, "Learning to generate synthetic data via compositing", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 461–470.

[9] F. Poucin, A. Kraus, and M. Simon, "Boosting instance segmentation with synthetic data: A study to overcome the limits of real world data sets", in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 945–953.

[10] R. Corvi, D. Cozzolino, G. Poggi, K. Nagano, and L. Verdoliva, "Intriguing properties of synthetic images: From generative adversarial networks to diffusion models", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 973–982.

[11] Q. Wang, J. Gao, W. Lin, and Y. Yuan, "Learning from synthetic data for crowd counting in the wild", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 8198–8207.

[12] A. Kortylewski, B. Egger, A. Schneider, T. Gerig, A. Morel-Forster, and T. Vetter, "Analyzing and reducing the damage of dataset bias to face recognition with synthetic data", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[13] Y. Zhang, D. Li, K. L. Law, X. Wang, H. Qin, and H. Li, "Idr: Self-supervised image denoising via iterative data refinement", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 2098–2107.

[14] R. Padilla, S. L. Netto, and E. A. Da Silva, "A survey on performance metrics for object-detection algorithms", in *2020 international conference on systems, signals and image processing (IWSSIP)*, IEEE, 2020, pp. 237–242.

[15] H. A. Alhaija, S. K. Mustikovela, L. Mescheder, A. Geiger, and C. Rother, "Augmented reality meets computer vision",

[16] R. Hess, *Blender foundations: The essential guide to learning blender 2.5*. Taylor & Francis, 2013.

[17] J. Shermeyer, T. Hossler, A. Van Etten, D. Hogan, R. Lewis, and D. Kim, "Rareplanes: Synthetic data takes flight", in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 207–217.

[18] E. Hatay, J. Ma, H. Sun, J. Fang, Z. Gao, and H. Yu, "Learning to detect phone-related pedestrian distracted behaviors with synthetic data", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2981–2989.

[19] M. Denninger, M. Sundermeyer, D. Winkelbauer, Y. Zidan, D. Olefir, M. Elbadrawy, A. Lodhi, and H. Katam, "Blenderproc", *arXiv preprint arXiv:1911.01911*, 2019.

[20] D. P. Kingma, M. Welling, *et al.*, "An introduction to variational autoencoders", *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019.

[21] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview", *IEEE signal processing magazine*, vol. 35, no. 1, pp. 53–65, 2018.

[22] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, Y. Shao, W. Zhang, B. Cui, and M.-H. Yang, "Diffusion models: A comprehensive survey of methods and applications", *arXiv preprint arXiv:2209.00796*, 2022.

[23] P. Xenopoulos, H. Doraiswamy, and C. Silva, "Valuing player actions in counter-strike: Global offensive", in *2020 IEEE international conference on big data (big data)*, IEEE, 2020, pp. 1283–1292.

[24] A. R. Sekkat, Y. Dupuis, P. Vasseur, and P. Honeine, "The omniscape dataset", in *2020 IEEE International conference on robotics and automation (ICRA)*, IEEE, 2020, pp. 1603–1608.

[25] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games", in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, Springer, 2016, pp. 102–118.

[26] 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.

[27] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, "Domain generalization: A survey", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[28] M. Huisman, J. N. Van Rijn, and A. Plaat, "A survey of deep meta-learning", *Artificial Intelligence Review*, vol. 54, no. 6, pp. 4483–4541, 2021.

[29] A. Farahani, S. Voghoei, K. Rasheed, and H. R. Arabnia, "A brief review of domain adaptation", *Advances in data science and information engineering: proceedings from ICDATA 2020 and IKE 2020*, pp. 877–894, 2021.

[30] Y. Yuan and M. Sester, "Comap: A synthetic dataset for collective multi-agent perception of autonomous driving", *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 43, pp. 255–263, 2021.

[31] S. Zhao, M. Gong, T. Liu, H. Fu, and D. Tao, "Domain generalization via entropy regularization", *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 096–16 107, 2020.

[32] Z. Marinov, A. Roitberg, D. Schneider, and R. Stiefelhagen, "Modselect: Automatic modality selection for synthetic-to-real domain generalization", in *European Conference on Computer Vision*, Springer, 2022, pp. 326–346.

[33] W. Chen, Z. Yu, Z. Wang, and A. Anandkumar, "Automated synthetic-to-real generalization", in *International Conference on Machine Learning*, PMLR, 2020, pp. 1746–1756.

[34] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques", *Global Transitions Proceedings*, vol. 3, no. 1, pp. 91–99, 2022.

[35] S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen, "Image data augmentation for deep learning: A survey", *arXiv preprint arXiv:2204.08610*, 2022.

[36] Y. Shi, X. Yu, K. Sohn, M. Chandraker, and A. K. Jain, "Towards universal representation learning for deep face recognition", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6817–6826.

[37] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples", *arXiv preprint arXiv:1412.6572*, 2014.

[38] R. Volpi, H. Namkoong, O. Sener, J. C. Duchi, V. Murino, and S. Savarese, "Generalizing to unseen domains via adversarial data augmentation", *Advances in neural information processing systems*, vol. 31, 2018.

[39] S. Shankar, V. Piratla, S. Chakrabarti, S. Chaudhuri, P. Jyothi, and S. Sarawagi, "Generalizing across domains via cross-gradient training", *arXiv preprint arXiv:1804.10745*, 2018.

[40] Y. Lu and J. Lu, "A universal approximation theorem of deep neural networks for expressing probability distributions", *Advances in neural information processing systems*, vol. 33, pp. 3094–3105, 2020.

[41] K. R. Shahapure and C. Nicholas, "Cluster quality analysis using silhouette score", in *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, 2020, pp. 747–748. DOI: 10.1109/DSAA49011.2020.00096.

[42] S.-i. Amari, "Backpropagation and stochastic gradient descent method", *Neurocomputing*, vol. 5, no. 4-5, pp. 185–196, 1993.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", *arXiv preprint arXiv:1412.6980*, 2014.

[44] M. Claesen and B. De Moor, "Hyperparameter search in machine learning", *arXiv preprint arXiv:1502.02127*, 2015.

[45] F. Pérez-Cruz, "Kullback-leibler divergence estimation of continuous distributions", in *2008 IEEE international symposium on information theory*, IEEE, 2008, pp. 1666–1670.

[46] G. R. Terrell and D. W. Scott, "Variable kernel density estimation", *The Annals of Statistics*, pp. 1236–1265, 1992.

[47] H. Cao, C. Tan, Z. Gao, G. Chen, P.-A. Heng, and S. Z. Li, "A survey on generative diffusion model", *arXiv preprint arXiv:2209.02646*, 2022.

[48] Z. Zhang, M. Li, and J. Yu, "On the convergence and mode collapse of gan", in *SIGGRAPH Asia 2018 Technical Briefs*, 2018, pp. 1–4.

[49] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of yolo algorithm developments", *Procedia Computer Science*, vol. 199, pp. 1066–1073, 2022.

[50] 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.

[51] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection", in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9627–9636.

[52] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology", *IEEE transactions on pattern analysis and machine intelligence*, no. 4, pp. 532–550, 1987.

[53] D. A. Reynolds, "Gaussian mixture models", in *Encyclopedia of Biometrics*, 2009. [Online]. Available: https://api.semanticscholar.org/CorpusID:1063711.

[54] M. Ahmed, R. Seraj, and S. M. S. Islam, "The k-means algorithm: A comprehensive survey and performance evaluation", *Electronics*, vol. 9, no. 8, p. 1295, 2020.

[55] W. Samek, T. Wiegand, and K.-R. Müller, "Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models", *arXiv preprint arXiv:1708.08296*, 2017.

[56] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning", *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[57] S. R. Searle, *Linear models*. John Wiley & Sons, 1997, vol. 65.

[58] L. Breiman, "Random forests", *Machine learning*, vol. 45, pp. 5–32, 2001.

[59] M. Cui *et al.*, "Introduction to the k-means clustering algorithm based on the elbow method", *Accounting, Auditing and Finance*, vol. 1, no. 1, pp. 5–8, 2020.

[60] T. Kailath, "The divergence and bhattacharyya distance measures in signal selection", *IEEE transactions on communication technology*, vol. 15, no. 1, pp. 52–60, 1967.

[61] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, "Neural style transfer: A review", *IEEE transactions on visualization and computer graphics*, vol. 26, no. 11, pp. 3365–3385, 2019.

[62] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "Knn model-based approach in classification", in *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings*, Springer, 2003, pp. 986–996.

[63] S. Weglarczyk, "Kernel density estimation and its application", in *ITM web of conferences*, EDP Sciences, vol. 23, 2018, p. 00 037.

[64] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context", in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer, 2014, pp. 740–755.