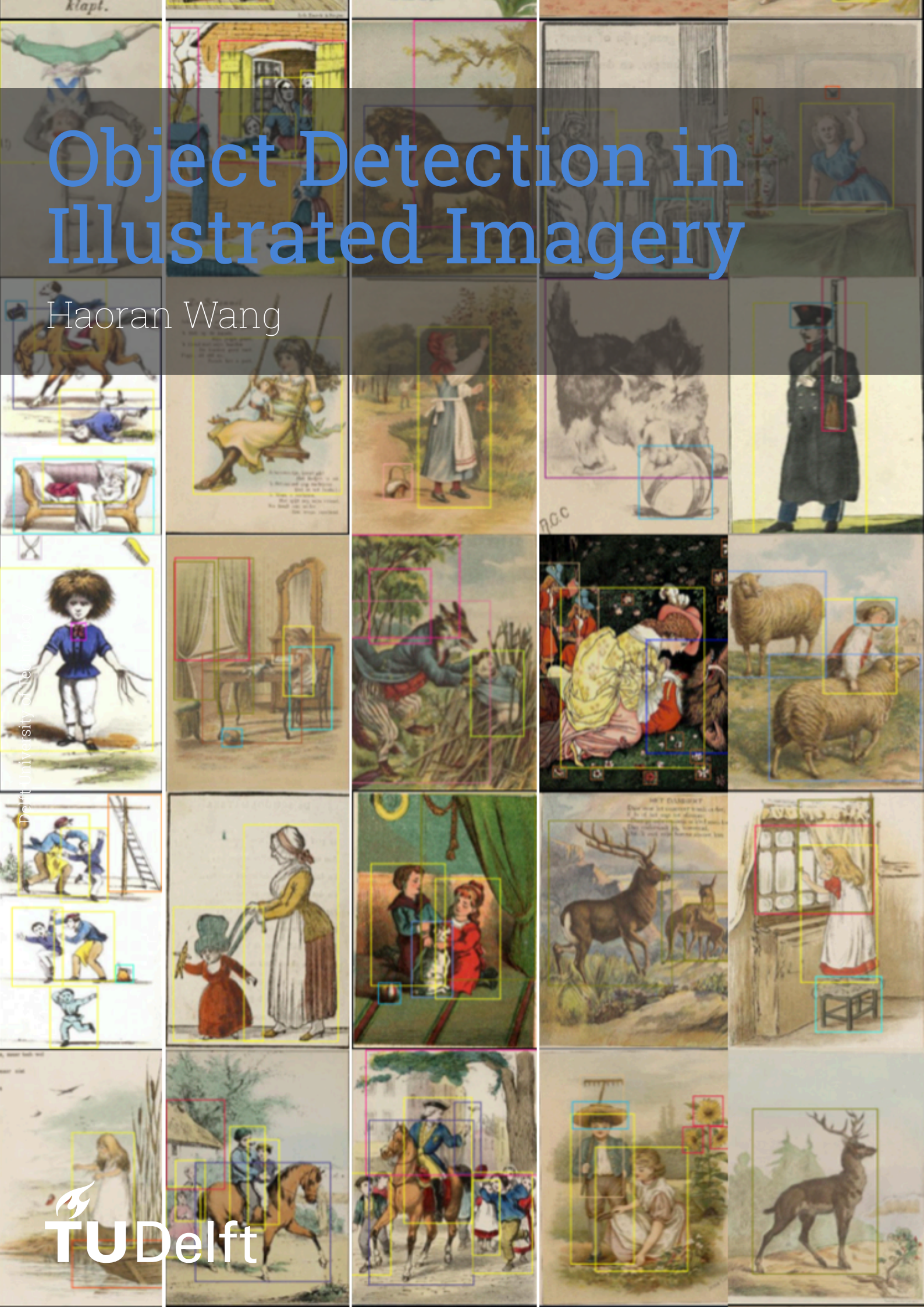


Object Detection in Illustrated Imagery

Haoran Wang

Delft University of Technology

TU Delft

Object Detection in Illustrated Imagery

by

Haoran Wang

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday August 21, 2023 at 15:00 PM.

Student Number: 5468175
Thesis Advisor: Jan van Gemert
Daily supervisor: Seyran Khademi
Project duration: November 14, 2022 – August 21, 2023
Faculty: Faculty of Electrical Engineering, Mathematics & Computer Science
Research Group: Pattern Recognition and Bioinformatics
Lab: Computer Vision
Thesis committee: Prof. Jan van Gemert
Prof. Seyran Khademi
Prof. Jie Yang

This thesis is confidential and cannot be made public until August 21, 2023.

TU Delft

Preface

This report presents the work of my master's thesis project on the topic *Object Detection in Illustrated Imagery*. The text is structured in three parts, an introduction, a scientific paper presenting our work in a compact manner using the format as in the Computer Vision and Pattern Recognition Conference (CVPR), and a supplement that introduces topics of the thesis and elaborates upon the scientific paper.

First, I would like to thank my daily supervisor, Seyran Khademi, for her invaluable guidance and feedback during the weekly meetings. Her expertise and support have been instrumental in shaping the work presented in this thesis. I would also like to thank my thesis supervisor, Jan van Gemert, for his valuable feedback and support. I would also like to extend my thanks to Jie Yang, for his interest in my thesis and evaluation of this work as a member of the thesis committee.

I take this opportunity to thank my parents and my sister, for their continuous support, and understanding throughout this journey. Their unwavering belief in me has been a source of motivation and strength. I would also like to thank my friends who have helped me in many forms.

Haoran Wang
Delft, August 2023

Contents

1

Introduction

Computer vision is a rapidly evolving field that aims to enable machines to understand and interpret visual data, emulating human vision capabilities. The core of computer vision research is the task of object detection, which involves identifying and localizing objects within images or videos. Object detection plays a crucial role in a wide range of applications, including autonomous driving, medical imaging, augmented reality, etc [1]. Advancements in deep learning and the availability of large-scale datasets have propelled significant progress in computer vision. State-of-the-art object detection models, such as Faster R-CNN [2], RetinaNet [3], and YOLO (You Only Look Once) [4], have gained remarkable performance improvements on benchmark datasets.

However, most computer vision research focuses on real images. Illustration images are rarely explored in computer vision. Exploring computer vision for illustrations has many potential applications, such as archive management for illustration books, and image understanding for graphic images.

To explore this field, we introduce a new dataset. The original dataset is Ot & Sien Dataset [5]. The purpose of this dataset is to help the development of automatic visual object detection in children's book illustrations. Mistakes, such as mislabelling, overlapped categories and images, and non-existing categories, have been corrected in the original dataset.

The dataset we proposed poses several challenges that need to be addressed. One of the challenges is the issue of imbalanced categories, where certain object categories may have a significantly larger number of instances compared to others. This imbalance can affect the performance of object detection models, as they may become biased toward the dominant categories while neglecting the minority ones. Additionally, the dataset exhibits a natural long-tail distribution, with some categories being rare or occurring infrequently. This further exacerbates the challenge of accurately detecting and classifying objects across the entire spectrum of the dataset.

Another challenge stems from the natural art diversity of the illustrations in the dataset. State-of-the-art object detection models, such as the YOLO model, have primarily been trained on high-quality photo datasets like MS COCO [6]. Unlike standardized photos found in conventional datasets, illustrations can vary greatly in style, artistic representation, and visual characteristics, and thus may not exhibit the same generalizability of texture information as seen in high-quality digital photos. This diversity introduces additional complexity to the object detection task.

Moreover, real photo-like datasets contain numerous categories that are irrelevant to the context of illustrations, requiring the fine-tuning of classification layers to adapt to custom datasets. In scenarios where prior information about the categories in the illustration is unavailable, few-shot learning has to be used. Besides, fine-tuning object detection models on real-photo datasets may prove to be challenging for this dataset. Given the differences in pixel representation and textual information between illustrations and photos, directly fine-tuning on real-photo datasets like COCO may not lead to optimal performance.

The rest of the report is structured as follows. Chapter 2 is a scientific paper that describes the thesis. Chapter 3 gives a technical background on the scientific paper, including deep learning, object detection, few-shot object detection, and more information about our dataset.

2

Scientific Paper

Object Detection for Illustrated Imagery

Haoran Wang
h.wang-46@student.tudelft.nl
Delft University of Technology

Abstract

In contrast to the prevalent focus on real photos in computer vision research, we present a contribution by making the Ot & Sien dataset [1] machine learning-ready for object detection tasks in illustrations. We refer to the new dataset as Ot & Sien++ that is composed of scanned images of children's book illustrations, thereby venturing into an unexplored domain. The primary objective of this research is to investigate the generalization capabilities of existing object detection models to this unique dataset and establish benchmarks for this dataset.

To evaluate the performance of existing object detection models on our proposed dataset, we employed the widely used YOLOv5 as a benchmark. To mitigate the inherent imbalance of the dataset, various data augmentation techniques were applied. The results demonstrated the effectiveness of the object detection model and data augmentation in the context of children's book illustrations. In addition, this research also explored applying few-shot learning models to the dataset. Baseline models were investigated to examine the potential of few-shot learning in the context of object detection in illustrations.

The proposed dataset elicits new challenges in object detection and will serve as a valuable resource for researchers in this domain. Our dataset can be found at https://data.4tu.nl/datasets/d1f3ca5c-f1e4-48f5-9a04-0564572d2b9c/1.

(a) Example images from the dataset

(b) gridded t-SNE visualization of part of the dataset

Figure 1. Dataset Visualizations

1. Introduction

Computer vision is a rapidly evolving field aimed at enabling machines to understand and interpret visual data and mimic human visual abilities. With recent advancements in deep learning and the availability of large-scale datasets, significant progress has been made in various computer vision tasks. One of the most developed tasks in computer vision research is object detection, which involves recognizing and localizing objects in an image or video. This task is of great importance in applications such as autonomous driving, medical imaging, and augmented reality. To tackle this challenge, state-of-the-art object detection models have emerged, including Faster R-CNN [2], RetinaNet [3], and YOLO (You Only Look Once) [4]. These models have achieved remarkable performance gains on benchmark datasets, pushing the boundaries of object detection capabilities.

However, most computer vision research focuses on photo imagery. Illustration images are rarely explored in

computer vision. The reasons can be attributed to the lack of comprehensive and well-annotated datasets specific to illustrations and the relatively lesser attention given to applications of object detection in illustrations compared to other mainstream computer vision tasks. Exploring computer vision for illustrations has many potential applications, such as archive management for illustration books, image understanding for graphic images as well as book summarization and quantitative research on archives.

To explore this field, we introduce a new dataset. The original dataset is Ot & Sien Dataset [1]. The purpose of this dataset is to help the development of automatic visual object detection in children's book illustrations. Mistakes, such as mislabelling, overlapped categories and images, and non-existing categories, have been corrected in the original dataset. To provide a glimpse of the dataset, example images are shown in Figure 1a. The properties of our new dataset are summarized as:

- The dataset consists of illustrations rather than photos.
- 1451 images with 8241 objects (5.7 per image) are annotated including the categories and bounding boxes.
- All images are resized to 416×416 with black fitting edges to adapt to the training procedure.
- The dataset has 164 classes and follows a natural long-tail property, with some object categories being rare.
- The dataset has imbalanced categories.

Due to the space limit, we present a partial visualization of the dataset using t-SNE [5] and then gridded in Figure 1b. The visualization provides a glimpse into the distribution and relationships among the data points. Specific clusters, such as people, animals, and various characters can be seen in the visualization, indicating inherent patterns within the dataset.

The dataset we proposed poses several challenges that need to be addressed. One of the challenges is the issue of imbalanced categories, where certain object categories may have a significantly larger number of instances compared to others. This imbalance can affect the performance of object detection models, as they may become biased toward the dominant categories while neglecting the minority ones. Additionally, the dataset exhibits a natural long-tail distribution, with most categories being rare or occurring infrequently. This further exacerbates the challenge of accurately detecting and classifying objects across the entire spectrum of the dataset.

Another challenge stems from the natural art diversity of the illustrations in the dataset. State-of-the-art object detection models, such as the YOLO model, have primarily been trained on high-quality photo datasets like MS COCO [6]. Unlike standardized photos found in conventional datasets, illustrations can vary greatly in style, artistic representation, and visual characteristics, and thus may not exhibit the same generalizability of texture information as seen in high-quality digital photos. This diversity introduces additional complexity to the object detection task.

Moreover, real photo-like datasets contain numerous categories that are irrelevant to the context of illustrations, requiring the fine-tuning of classification layers to adapt to custom datasets. In scenarios where prior information about the categories in the illustration is unavailable, few-shot learning has to be used. Besides, fine-tuning object detection models on real-photo datasets may prove to be challenging for this dataset. Given the differences in pixel representation and textual information between illustrations and photos, directly fine-tuning on real-photo datasets like COCO may not lead to optimal performance.

We acknowledge the scarcity and imbalance of the dataset as a realistic scenario. We list the key contributions as follows:

- Dataset Preparation: We have made the Ot & Sien dataset ready for object detection task, enabling further research in computer vision specifically tailored to illustrations.
- Baseline Model Evaluation: We have conducted a comprehensive evaluation of different baseline object detection models on our dataset.
- Generalization Capability Analysis: We have investigated the generalization capability of object detection models trained on real-photo datasets when applied to our dataset.
- Data Augmentation Effectiveness: We have explored the effectiveness of various data augmentation techniques on our dataset.

2. Related Work

2.1. Datasets

Several benchmark datasets have played a significant role in advancing research for various tasks, such as Pascal VOC [7], MS COCO (Microsoft Common Objects in Context) [6], and LVIS (Large Vocabulary Instance Segmentation) [8]. These datasets have provided standardized benchmarks and large-scale labeled data, allowing researchers to evaluate and compare the performance of different computer vision models.

The Pascal VOC dataset has been a benchmark in the computer vision community for many years. It includes diverse images annotated with object bounding boxes and pixel-level semantic masks. The dataset covers 20 object categories, including common categories such as cars, people, and animals. Pascal VOC has been used extensively for tasks such as object detection, semantic segmentation, and image classification. However, one limitation of PASCAL VOC is its relatively small size, consisting of around 10000 images in total.

MS COCO is a larger dataset that provides a more ex-

tensive and challenging collection of images compared to Pascal VOC. It contains over 200000 images with pixel-level annotations for objects. The dataset contains 80 object categories and covers a wide range of complex scenes and object instances. The dataset's rich annotations and large-scale nature have made it a crucial resource for training and evaluating object detection algorithms.

LVIS is a relatively new dataset that focuses on instance segmentation. It included a larger vocabulary of 1200 object categories and provided around 2 million high-quality instance segmentation masks for 164k images. LVIS has a naturally long tail distribution, providing a more comprehensive representation of real-world object frequencies. This dataset focuses on evaluating object detection models on rare and low-frequency categories.

There are several existing illustration datasets. For example, DanbooRegion [9] is an illustration region dataset that contains a large number of artistic region compositions paired with corresponding cartoon illustrations. WaterColor2k [10] is a dataset used for cross-domain object detection which contains 2k watercolor images with image and instance-level annotations. They present a new framework for cross-domain weakly supervised object detection, which tries to adapt pre-trained source domain knowledge to a new domain.

2.2. Object Detection

Object detection is a well-studied problem in computer vision, and various approaches have been proposed to tackle it effectively. Modern object detection methods can be categorized into two main types: one-stage methods and two-stage methods.

One of the most influential two-stage object detection frameworks is the Region-based Convolutional Neural Network (R-CNN) family of methods [2, 11, 12]. These methods propose region proposal techniques to generate candidate object regions and employ convolutional neural networks (CNNs) to classify and localize objects within these regions. These approaches have demonstrated impressive performance on various benchmark datasets.

Faster R-CNN [2], proposed in 2015, revolutionized object detection by introducing a two-stage framework that combines region proposal generation and object classification. The primary motivation behind Faster R-CNN was to alleviate the shortcomings of earlier approaches that relied on time-consuming external region proposal methods, such as Selective Search. It consists of a region proposal network (RPN) and a subsequent object detection network. The RPN shares convolutional layers with the detection network, allowing for efficient computation and reuse. This design ensures end-to-end training and enables the network to learn powerful representations for both region proposal and object classification.

However, the two-stage approaches mentioned above suffer from increased computational complexity due to the need for region proposal techniques. This limitation makes them less suitable for real-time applications where efficiency is a critical factor.

In contrast, one-stage approaches, such as the You Only Look Once (YOLO) series [4, 13, 14], adopt a unified approach that directly predicts object bounding boxes and class probabilities in a single pass through the network. It divides the input image into a grid and assigns each grid cell responsibility for detecting objects. YOLO is known for its fast inference speed, thus is suitable for real-time object detection applications. However, it may struggle with detecting small objects and precise localization due to its grid-based nature.

2.3. Imbalanced Dataset

In recent years, the issue of class imbalance in object detection datasets has gained significant attention. Imbalanced datasets, where certain object categories are underrepresented compared to others, can lead to biased model training and sub-optimal performance in minority classes.

To tackle this challenge, researchers have proposed various methods to mitigate the negative impact of class imbalance. One such method is the focal loss introduced by Lin *et al*. [3]. The focal loss addresses the class imbalance problem by assigning higher weights to misclassified examples from the minority classes during training. By giving more importance to the underrepresented classes, the model can focus on learning their distinctive features and improving their detection performance.

Data-level approaches also play a crucial role in addressing class imbalance [15]. Data augmentation techniques, such as rotation, scaling, flipping, or adding noise can help increase the diversity and quantity of data for underrepresented classes. This can help improve the model's ability to learn from and generalize to these classes.

Sampling techniques, such as random over-sampling (ROS) [16], random under-sampling (RUS) [16], and synthetic minority oversampling technique (SMOTE) [17], can balance the class distribution [15]. ROS replicates or generates new samples from the minority class, while RUS reduces the number of samples from the majority class. SMOTE generates synthetic samples by interpolating between existing minority class samples. These techniques aim to create a more balanced representation of all classes and alleviate the bias towards the majority classes.

Another effective approach is transfer learning [18], which leverages pre-trained models on large-scale datasets, such as ImageNet [19] or COCO, to initialize the object detection model. By transferring the learned features and knowledge from the pre-trained model, the model can benefit from the generalization and discriminative power of the

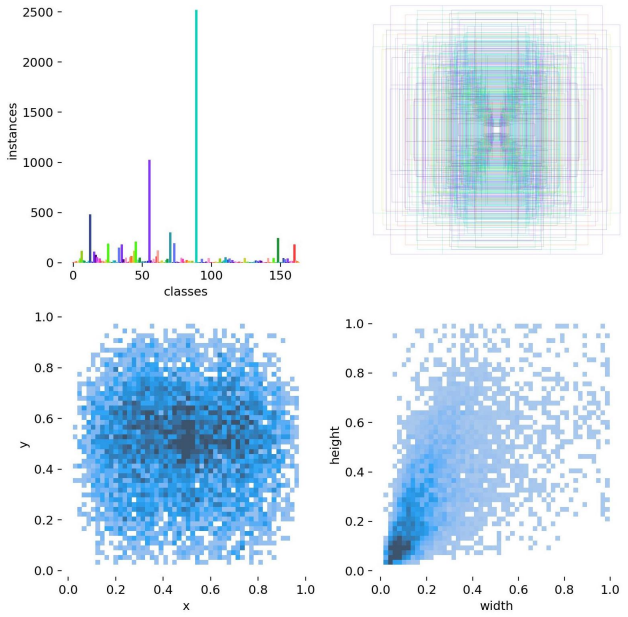


Figure 2. Visualization of image labels

large dataset. This can help mitigate the effects of class imbalance by providing a strong starting point for training on the imbalanced dataset.

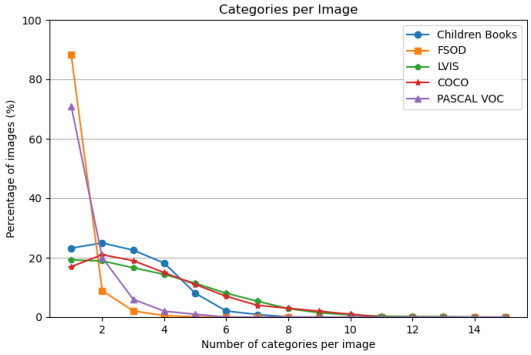
2.4. Few-shot Object Detection

Few-shot object detection extends the concept of few-shot learning to the domain of object detection, where the goal is to detect objects in images with only limited labeled examples.

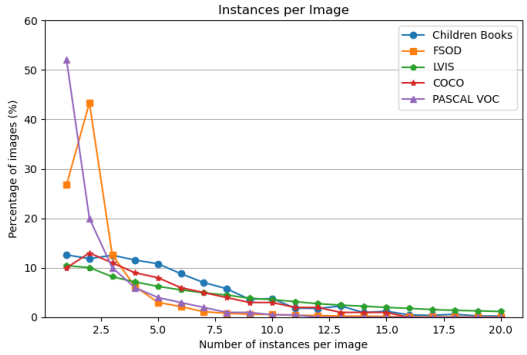
One approach to few-shot object detection is to leverage few-shot learning methods and adapt them to the object detection setting. For instance, Yan *et al*. [20] extends Faster/Mask R-CNN [2, 21] by proposing meta-learning over RoI (Region of Interest) instead of a full image. The core idea is to separate the complex information of multiple objects merged with the background. This enables Faster/Mask R-CNN [2,21]to function as a meta-learner for performing various tasks.

Besides, Kang *et al*. [22] propose a method where feature re-weighting schemes are incorporated into a single-stage object detector, specifically YOLOv2 [13]. This is achieved by employing a meta-learner that takes both the support images (a limited number of labeled images belonging to the novel/base classes) and the corresponding bounding box annotations as inputs.

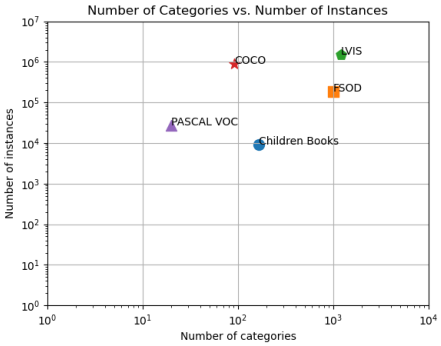
Furthermore, recent advancements in pre-training models have also been explored in few-shot learning. These models are pre-trained on large-scale datasets and then fine-tuned on few-shot learning tasks, enabling them to capture rich representations that can generalize well to new classes



(a) Distribution of categories per image



(b) Distribution of instances per image



(c) Number of categories *vs*. Number of instances

Figure 3. Dataset comparison

with limited labeled examples. Wang *et al*. [23] found that fine-tuning only the last layer of existing detectors on rare classes is crucial to the few-shot object detection task.

Despite the progress made in few-shot object detection, it remains an active research area with ongoing efforts to develop more effective algorithms.

3. Dataset

3.1. Dataset Statistics

There are 164 categories present in the 1451 children's book images, which have the same size 416×416. On average, each image is annotated with 5.7 objects.

	train	valid	test
No. Image	961	259	231
No. Box	5566	1459	1216
Avg No.Box/Img	5.79	5.63	5.26
Box Size (pixels)	[85, 167232]	[125, 161540]	[144, 150423]
Box Area Ratio	[0.00049, 0.966]	[0.00083, 0.869]	[0.00072, 0.933]
Box W/H Ratio	[0.0840, 7.769]	[0.0873, 6.310]	[0.0603, 7.643]

Table 1. Data splits

To understand the distribution of object instances across the different categories, the first figure in Figure 2 displays the instances-per-class distribution. This distribution highlights the inherent imbalance and low-shot nature of the dataset, with certain categories having a significantly larger number of instances compared to others.

The second figure in Figure 2 shows the shape and location of all bounding boxes in our dataset and the third figure shows the distribution of the center of all bounding boxes. The visualization demonstrates the spatial distribution of objects and illustrates that many bounding boxes are centered around the middle of the image.

Lastly, the size distribution of the bounding boxes is depicted in the last figure of Figure 2. This visualization illustrates that the dataset contains a significant number of small objects, as indicated by the higher density of data points in the lower region of the size distribution plot.

3.2. Datasets Comparison

We compare our dataset with four commonly used datasets in object detection and few-shot object detection, Pascal VOC, MS COCO, LVIS, and few-shot object detection (FSOD) [24].

The distribution of categories-per-image is depicted in Figure 3a, revealing that the majority of images contain 1 to 5 categories. Similarly, Figure 3b illustrates the distribution of instances-per-image, indicating that most images contain 1 to 8 instances. Our dataset exhibits a distribution that closely resembles that of the COCO dataset. The relationship between the number of categories and the number of instances is demonstrated in Figure 3c. Compared to the PASCAL VOC dataset, our dataset contains a comparable number of instances. However, it contains a larger number of categories. On the other hand, when compared to the COCO dataset, our dataset exhibits a similar number of categories, but significantly fewer instances. This discrepancy can be attributed to the relatively smaller size and long-tail distribution of our dataset.

4. Benchmarks and Experiments

In this section, we provide baseline results for our proposed dataset. We perform object detection using the most widely used YOLOv5 model and train the TFA model [23] as a few-shot learning baseline.

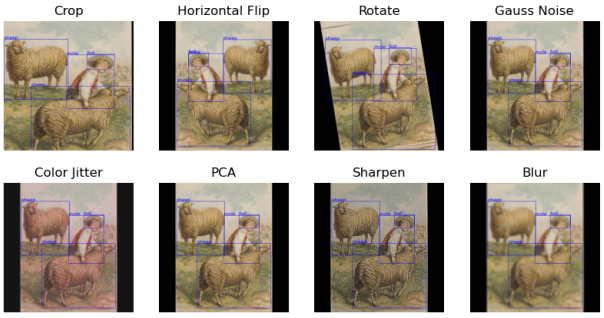


Figure 4. Data augmentation techniques

4.1. Object Detection

Dataset Preparation We split the dataset into train, validation, and test sets. Statistics can be found in Table 1.

Given the presence of imbalanced categories within the dataset, it is essential to address the inherent challenges associated with evaluating results on classes with limited instances. To ensure a fair and meaningful assessment, we adopted a categorization strategy to divide the dataset into two distinct groups: frequent categories and rare categories. The categorization was based on the abundance of instances within each class specifically within the training dataset. Categories with more than 40 instances in the training dataset were classified as frequent categories, whereas those with fewer instances were categorized as rare categories. This division allows us to focus our evaluation primarily on the frequent categories, which exhibit a higher representation within the dataset.

Experiments It has been proven that pre-training the model on a large-scale dataset, such as ImageNet and MS COCO, improves generalization and helps to prevent overfitting [25]. Therefore, we used the model weights learned

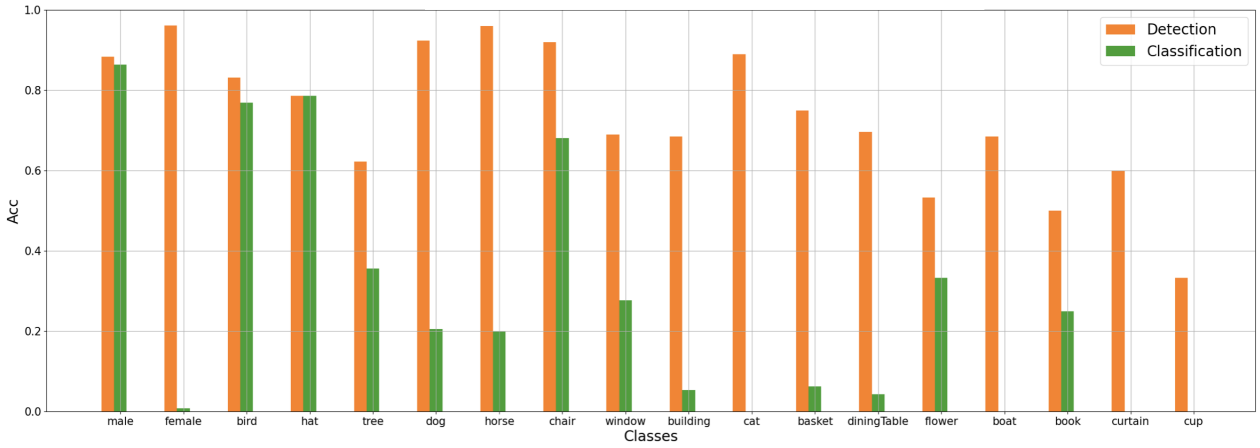


Figure 5. Detection and classification accuracy on the dataset w/o augmentation

from COCO and then fine-tune it on our dataset in the experiments.

In order to validate the efficacy of data augmentation on our dataset, we conducted experiments employing various augmentation strategies. It was demonstrated that a straightforward copy-paste technique can lead to improved performance [26]. Building upon this insight, we further explored the potential of common data augmentation techniques, including cropping, rotation, flipping, and other transformations. Augmentation techniques have been used are shown in Figure 4, which include crop, horizontal flip, rotate, gauss noise, color jitter, PCA, sharpen, and blur.

Given the inherent class imbalance present in our dataset, it is important to address this challenge during data augmentation. Conventional image-level augmentation techniques, while effective in introducing diversity, do not adequately address the imbalance issue and may even exacerbate it further. In light of this, we put forth a novel approach that focuses on bounding-box level augmentation, aiming to mitigate the imbalance problem while enhancing the dataset's richness and variability.

Our proposed bounding box level augmentation strategy operates by considering individual bounding boxes rather than the entire image. By doing so, we can selectively augment specific object instances, thereby offering a more targeted and controlled augmentation process. This approach enables us to maintain a better balance between different object categories, ensuring that each class receives appropriate augmentation, irrespective of its initial representation in the dataset. Examples are shown in Figure 6, where bears and pigs are being augmented.

Results The experimental results are presented in Table 2. The results demonstrate the positive impact of data augmentation on improving the model's performance on frequent classes and it also shows the generalization ability of the YOLOv5 model on our dataset. Furthermore, our

(a) Example 1: Bears (b) Example 2: Pigs

Figure 6. Examples of bounding-box level augmentation

novel bounding-box level augmentation approach exhibited promising results, as observed in the table.

	$mAP50_f$	$mAP75_f$	mAP_f
w/o augmentation	17.39	7.87	8.87
copy & paste	18.43	8.35	9.42
bbox augmentation	25.57	15.31	15.23
normal augmentations	32.03	17.84	18.13

Table 2. Object detection performance of YOLOv5 on frequent classes. w/o augmentation: original dataset without augmentation; copy & paste: simply copy and paste images; normal augmentations: uses image-level augmentation techniques; bbox augmentation: uses bounding-box level augmentation

Object detection models consist of two stages: detection, which localizes objects in an image, and classification, which assigns a specific class label to each detected object. To pinpoint the potential bottleneck in performance, we separate the evaluation of detection accuracy and classification accuracy. This allows us to analyze and assess the model's performance in each stage independently.

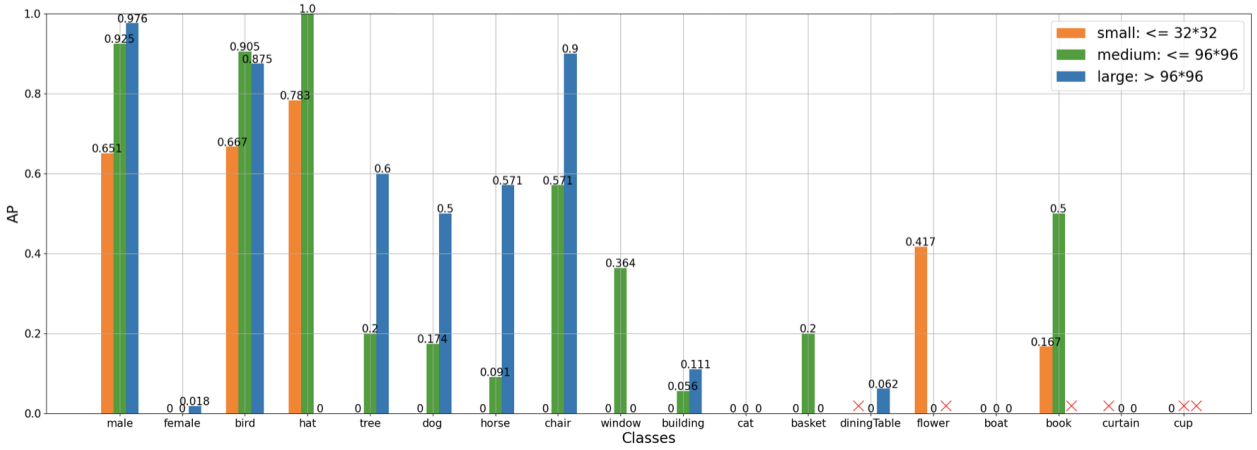


Figure 7. Performance on different sized objects

In Figure 5, we present the detection and classification accuracies on the original dataset without any augmentation. It is evident that the detection accuracies are generally higher than the classification accuracies. One notable observation is the extremely low classification accuracy for the "female" class. This discrepancy can be attributed to the significant class imbalance between the "male" and "female" classes. Given that the "male" class has a substantially larger number of instances compared to the "female" class, the model tends to predict all instances as "male" to optimize overall accuracy. This imbalance in training data distribution poses a challenge for the model to accurately classify "female" samples, resulting in near-zero classification accuracy for this class.

The complete results for detection and classification accuracies are provided in Table 3. It is evident from the table that in all cases, the localization accuracies are consistently higher than the classification accuracies. This observation suggests that the bottleneck in the model lies in the classification stage, as the model struggles to accurately classify objects despite successfully localizing them.

	Localization	Classification
w/o augmentation	73.58	27.14
copy & paste	73.27	31.05
bbox augmentation	75.76	38.77
normal augmentations	73.28	49.22

Table 3. Detection and classification accuracy on frequent classes

To investigate the model's performance on different sizes of bounding boxes, we conducted an evaluation based on the bounding box sizes. We divided the bounding boxes into three categories: small (s) with a pixel area of 32^2 or smaller, middle (m) with a pixel area between 32^2 and 96^2, and large (l) with a pixel area larger than 96^2.

	$mAP50_s$	$mAP50_m$	$mAP50_l$
w/o augmentation	14.92	23.70	25.63
copy & paste	15.88	30.32	33.50
bbox augmentation	17.88	31.95	52.30
normal augmentations	23.07	34.41	66.75

Table 4. Object detection performance by bounding box size on frequent classes

Figure 7 provides a visualization of the model's performance based on the bounding box size, using the original dataset without any augmentation. In the figure, red crosses indicate the absence of objects of a certain size. For example, there are no large-sized "cup" instances in the dataset. From the figure, we can observe that the model demonstrates overall better performance on middle and large-sized bounding boxes.

The complete results are presented in Table 4. From the table, we can observe that the model achieves better performance on middle-size and large-size bounding boxes compared to small-size bounding boxes. This observation can be attributed to the inherent characteristics of the YOLO model, which tends to struggle to accurately predict small objects.

4.2. Few-shot object detection

Dataset Preparation Following the few-shot learning evaluation metrics, we divided the dataset into two categories: base classes and novel classes. Base classes refer to classes with instance numbers larger than 40, while novel classes correspond to classes with instance numbers less than 40 and more than 20 to insure validity.

To assess the performance of the few-shot object detec-

Backbone	shots	mAP_{50}	mAP_{75}	mAP	mAP_s	mAP_m	mAP_l
R50	1	3.95	1.33	1.61	0.31	1.71	2.82
R50	2	4.32	2.12	2.46	1.21	2.59	3.58
R50	3	6.76	2.37	3.44	1.22	3.02	6.08
R50	5	7.28	2.93	4.06	2.11	3.39	6.67
R50	10	8.23	4.32	5.09	2.75	4.60	7.92
R101	1	4.07	2.36	2.52	0.52	2.74	4.31
R101	2	5.84	3.20	3.49	1.84	3.52	5.12
R101	3	6.85	3.43	4.12	2.09	4.10	6.17
R101	5	8.13	3.68	4.74	2.64	4.63	6.94
R101	10	9.41	4.69	5.52	3.10	5.15	8.31

Table 5. Results of few shot object detection

tion model across different shot scenarios, we created separate datasets containing 1, 2, 3, 5, and 10 shots of both base classes and novel classes. This allowed us to evaluate how the model's performance varied with the number of available training examples.

Experiments We adopt the two-stage fine-tuning approach (TFA) proposed in [23]. This approach involves two steps: initial training on the base classes and subsequent fine-tuning on a smaller balanced training set. Specifically, we train the entire object detector on the data-abundant base classes initially. Then, we fine-tune only the last layers of the detector on a small balanced training set.

For our implementation, we utilize the Detectron2 framework [27], which provides a robust and efficient platform for object detection tasks. The experiments in this study involved evaluating two models with different backbone architectures: ResNet-101 and ResNet-50. These models are variants of the ResNet [28] architecture, which is a popular choice for deep convolutional neural networks in computer vision tasks. The ResNet-101 model utilizes a ResNet backbone with 101 layers, offering a deeper and more complex network architecture.

The first model, Faster R-CNN R50-C4, employs a ResNet-50 backbone. The second model, Faster R-CNN R101-FPN, utilizes a ResNet-101 and FPN backbone. These models serve as our baseline models for performance evaluation.

Results Results of the few-shot object detection experiments are presented in Table 5. For validity, three sets of experiments and then calculate the average value of the results. Additionally, we assess the model's performance based on the size of the bounding boxes.

Upon examining the table, it is evident that the model with the ResNet-101 backbone achieves better overall performance, which aligns with our expectations. Furthermore, the model exhibits improved performance on middle and large-size objects compared to small-size objects.

It is important to note that due to the few-shot nature of the task, the performance on novel classes remains relatively low. However, with further refinements and adjustments, there is potential for enhancing the model's ability to generalize and detect objects from novel classes.

5. Conclusion

In conclusion, this study introduced a valuable dataset comprising children's book illustrations and conducted baseline experiments to explore object detection in this domain. This dataset presents new research challenges in the field of object detection. The unique characteristics of children's book illustrations, such as diverse artistic styles, varying object sizes, and imbalanced class distributions, require tailored approaches for accurate and robust detection.

The conducted baseline experiments provide valuable insights into the transferability of pre-trained models and the effectiveness of data augmentation techniques in this domain. However, further research is needed to address the specific challenges posed by this dataset.

Considering future research directions, it is essential to investigate advanced architectures tailored specifically for children's book illustrations. Developing specialized backbone architectures or incorporating attention mechanisms could further improve the models' performance by capturing the unique visual patterns and characteristics present in our dataset. The findings and considerations presented here pave the way for advancements in automated analysis and understanding of visual content in the domain of children's book illustrations.

References

[1] S. Khademi, S. Veldhoen, and L. Wilms. Ot sien dataset. https://lab.kb.nl/dataset/ot-sien-dataset, 2021. KB Lab: The Hague. 1, 2

[2] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1, 3, 4

[3] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 1, 3

[4] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1, 3

[5] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 2

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence 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*, pages 740–755. Springer, 2014. 2

[7] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88:303–338, 2010. 2

[8] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019. 2

[9] Lvmin Zhang, Yi JI, and Chunping Liu. Danbooregion: An illustration region dataset. In *European Conference on Computer Vision (ECCV)*, 2020. 3

[10] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5001–5009, 2018. 3

[11] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 3

[12] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 3

[13] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 3, 4

[14] Glenn Jocher et. al. ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support, October 2021. 3

[15] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):1–54, 2019. 3

[16] Joffrey L Leevy, Taghi M Khoshgoftaar, Richard A Bauder, and Naeem Seliya. A survey on addressing high-class imbalance in big data. *Journal of Big Data*, 5(1):1–30, 2018. 3

[17] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority oversampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002. 3

[18] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010. 3

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3

[20] Xiaopeng Yan, Ziliang Chen, Anni Xu, Xiaoxi Wang, Xiaodan Liang, and Liang Lin. Meta r-cnn: Towards general solver for instance-level low-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9577–9586, 2019. 4

[21] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 4

[22] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8420–8429, 2019. 4

[23] Xin Wang, Thomas E Huang, Trevor Darrell, Joseph E Gonzalez, and Fisher Yu. Frustratingly simple few-shot object detection. *arXiv preprint arXiv:2003.06957*, 2020. 4, 5, 8

[24] Qi Fan, Wei Zhuo, Chi-Keung Tang, and Yu-Wing Tai. Few-shot object detection with attention-rpn and multi-relation detector. In *CVPR*, 2020. 5

[25] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pages 2712–2721. PMLR, 2019. 5

[26] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2918–2928, 2021. 6

[27] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 8

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 8

3

Supplement

Technical explanations of core concepts used in the scientific article of part 2.

3.1. Deep Learning

Deep learning, a sub-field of machine learning, has become a powerful and transformative artificial intelligence approach. It utilizes multiple-layer neural networks to learn intricate patterns and representations from vast amounts of data. With its great ability in discovering hierarchical features and extract meaningful insights, deep learning has demonstrated remarkable success across various domains, including natural language processing, computer vision, speech recognition, etc.

Deep learning includes a wide range of machine learning techniques that use multiple-layer neural networks to learn intricate patterns and representations from data. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) are two common deep learning designs that have demonstrated exceptional performance in computer vision and sequential data analysis applications, respectively. However, at the heart of deep learning is the fundamental building element known as multi-layer perception (MLP).

3.1.1. Multi-layer Perceptron

Deep learning is based on artificial neural networks inspired by the structure and function of the human brain. These neural networks consist of interconnected layers of artificial neurons, each performing simple computations on input data and passing the results to the next layers. The basic structure of a neural network is shown in Figure 3.1.

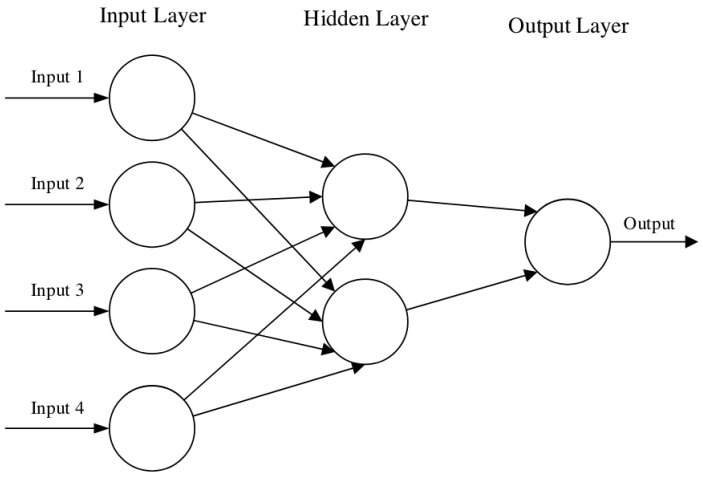


Figure 3.1: Basic Structure of Neural Network [7, Fig. 1]

To have a better understanding of how a neural network works, take the example of a feed-forward network, which is the basic building block of deep learning models. A simple feed-forward network, also known as a Multi-Layer Perceptron (MLP), accepts an input x and allows information to pass through its intermediate layers to produce some output y. In other words, it approximates some function f so that the output $y = f(x)$ is close to the predicted output y.

Through an iterative training process, deep learning models learn to recognize and generalize patterns by adjusting the weights and biases associated with each neuron. The depth and complexity of these neural networks allow them to capture complex relationships and representations that are difficult to achieve with traditional machine learning methods.

One of the key strengths of deep learning is to learn directly from raw data without manual feature engineering. This data-driven approach opens up new possibilities for solving complex problems in areas such as image classification, object detection, language translation, and medical diagnosis.

3.1.2. Activation Functions

Activation functions are often used in neural networks to add complexity. It introduces non-linearity to the network, allowing it to learn and approximate complex relationships within the data. Common activation functions include the Sigmoid function, rectified linear unit (ReLU), etc. The Sigmoid function restricts the output to the interval $[0, 1]$. The ReLU function is the most commonly used function, which simply computes the $max(x, 0)$, squishing all negative inputs to zero and keeping positive values unchanged. Sigmoid and ReLU activations are visualized in Figure 3.2 and 3.3.

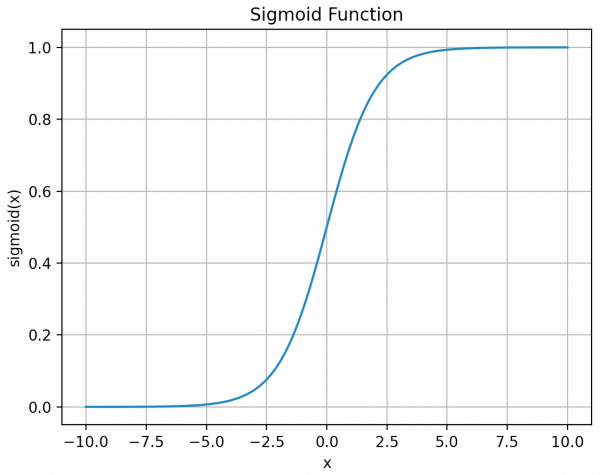


Figure 3.2: Sigmoid Function

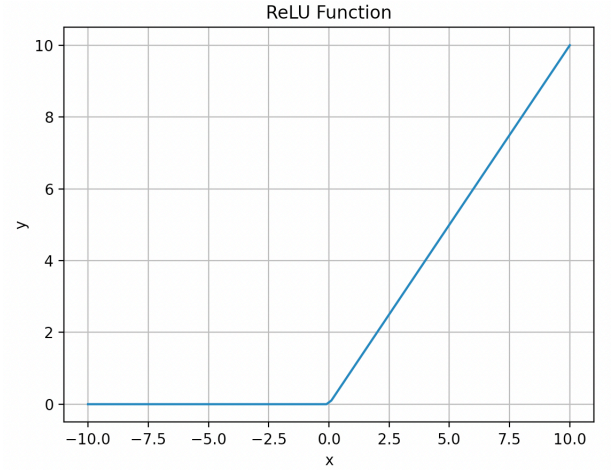


Figure 3.3: ReLU Function

3.1.3. Training a Neural Network

In simple words, training a neural network is to find a function that generates a $\hat{y}$ that is closest to the actual expected output y. The neural network must find the best set of learnable parameters for the weights and biases in order to get the best approximation. These parameters are usually initialized randomly in the beginning.

However, training deep neural networks requires massive computing resources, often relying on specialized hardware such as graphic processing units (GPUs) or tensor processing units (TPUs). Furthermore, the large amount of labeled data required for efficient training can create a bottleneck in some domains. Overcoming these challenges requires expertise in model architecture design, optimization techniques, regularization methods, and data augmentation strategies.

As deep learning research advances, new architectures such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformers continue to improve the performance of various applications.

3.1.4. Convolutional Neural Network

Convolutional Neural Networks (CNNs) are a type of artificial neural network designed specifically for processing and interpreting structured grid-like data such as photos, and video. It is particularly effective in analyzing visual data due to its ability to capture spatial information and learn local patterns through the use of convolutional layers. CNNs have transformed the field of computer vision, excelling in tasks such as image classification, object detection, and semantic segmentation.

CNNs are comprised of three types of layers. These are convolutional layers, pooling layers, and fully-connected layers. A simple CNN architecture for MNIST classification is illustrated in Figure 3.4.

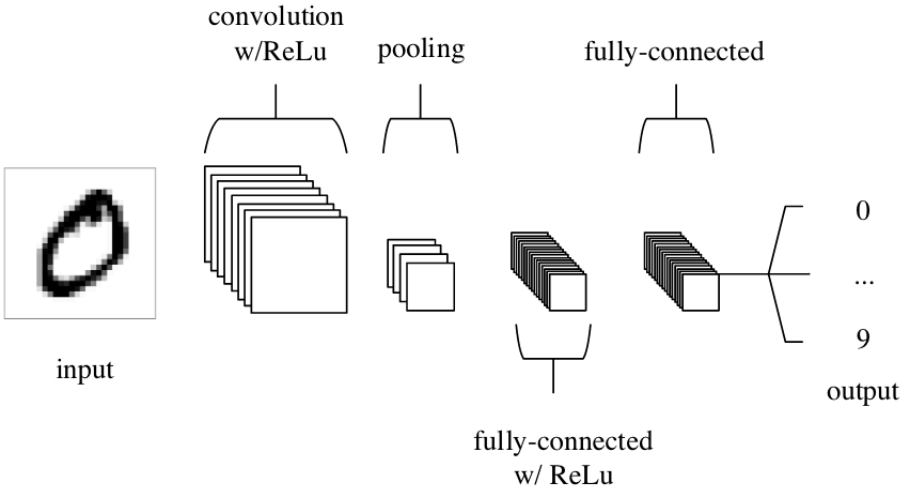


Figure 3.4: An simple CNN architecture, comprised of just five layers [7, Fig. 2]

Convolutional layers are at the heart of CNNs. These layers are made up of filters or kernels that perform convolutions on the input data to extract local patterns and features. Convolution involves sliding a set of small filters (also known as kernels or feature detectors) over the input, computing the dot product between the filter and local patches of the input.

Pooling layers are often inserted after convolutional layers to reduce the spatial dimensions of the data and retain the most salient features. They aggregate information from a local neighborhood of the input, typically through operations such as max pooling or average pooling. The main purpose of pooling layers is to down-sample the feature maps, reducing their size and computational complexity while capturing the most relevant and salient features.

A max pooling operation is visualized in 3.5. Max pooling is a commonly used pooling operation, where the maximum value within each pooling window is selected and retained, discarding the rest of the values. This helps capture the most prominent feature within each local neighborhood. On the

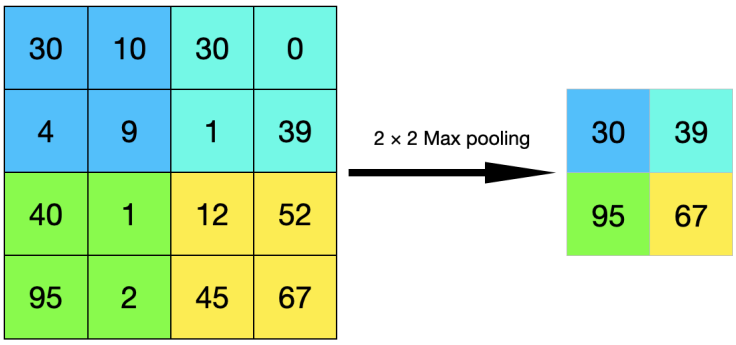


Figure 3.5: A max pooling operation. The maximum value of each region, denoted by a distinct color, is the output.

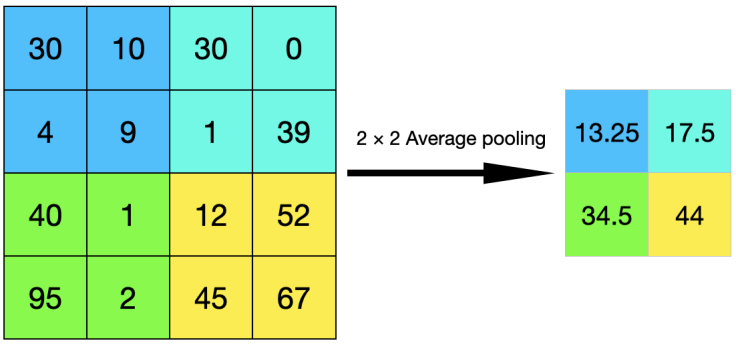


Figure 3.6: An average pooling operation. The average value of each region denoted by a distinct color, is the output.

other hand, an average pooling operation is visualized in 3.6. Average pooling computes the average value within each pooling window, which can be useful for capturing general information about the local neighborhood.

However, it's worth noting that pooling operations can result in information loss since they discard some of the detailed information within the local neighborhoods. This can be mitigated to some extent by using smaller pooling window sizes or by using techniques like strided pooling, which applies the pooling operation with a larger stride to reduce downsampling.

The output of the convolutional and pooling layers is typically flattened and fed into fully connected layers. These layers are similar to those in traditional artificial neural networks and are responsible for performing classification or regression tasks.

3.2. Object Detection

3.2.1. Definition

Computer vision involves various visual recognition tasks, including image classification, object detection, instance segmentation, and semantic segmentation, whose differences are described in Figure 3.7.

Specifically, image classification involves identifying the objects' semantic categories in an image, while object detection also predicts the location of each object with a bounding box. Semantic segmentation assigns a specific category label to each pixel. However, semantic segmentation does not differentiate between multiple objects of the same category. Instance segmentation, which combines object detection and semantic segmentation, identifies different objects and assigns them separate categorical pixel-level masks.

Object detection is the basic step towards many different computer vision tasks, such as face recognition, pedestrian recognition, and video analysis. Modern object detection methods can be categorized into two main types: one-stage methods and two-stage methods.

Two-stage methods follow the traditional object detection pipeline by generating region proposals

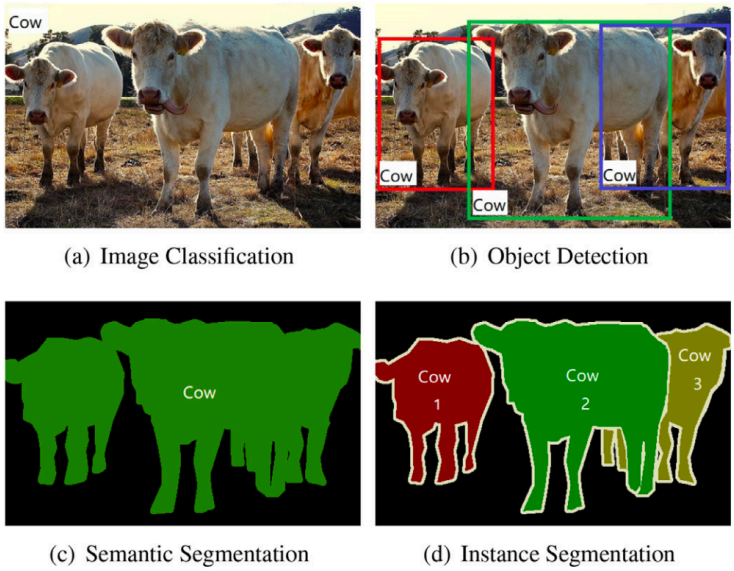


Figure 3.7: Comparison of different visual recognition tasks in computer vision [8, Fig. 1]

and classifying each proposal into different object categories. Two-stage methods include R-CNN [9], Fast R-CNN [10], Faster R-CNN [2], Mask R-CNN [11], etc.

3.2.2. R-CNN

R-CNN (Region-based Convolutional Neural Network) is an object detection framework introduced by Girshick et al. [9] in 2014. It changed the field of object detection by combining the power of deep learning with region proposal techniques. The framework is visualized in 3.8.

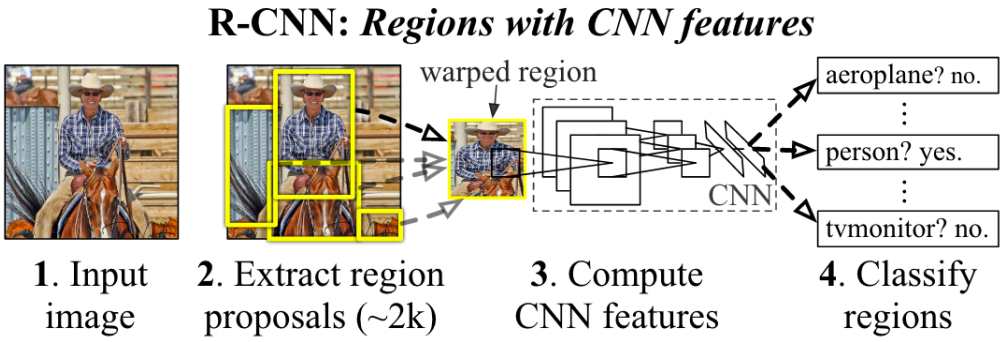


Figure 3.8: R-CNN model [9, Fig. 1]

1. Region Proposal: Initially, around 2000 bottom-up region proposals are extracted using selective search or a similar algorithm. These regions are likely to contain objects of interest.
2. CNN Feature Extraction: Region proposals are resized to a fixed size and fed into a convolutional neural network to extract rich feature representations. The network is typically pre-trained on a large-scale image classification task, such as ImageNet, to learn generic visual features.
3. Object Classification: The extracted features from each region proposal are forwarded to additional layers, including fully connected layers, to perform object classification. This step involves predicting the presence of an object and assigning it to one of the predefined classes.
4. Bounding Box Regression: In addition to object classification, R-CNN also performs bounding box regression to refine the localization of objects. It predicts the coordinates of a tighter bounding box around the detected object, aiming for more accurate object localization.

5. Non-Maximum Suppression: To address the issue of duplicate detection, a post-processing step called non-maximum suppression (NMS) is applied. NMS eliminated redundant bounding boxes by keeping only the most confident ones.

R-CNN made significant advancements in object detection, but it also has several limitations. R-CNN uses selective search or similar algorithms to generate region proposals for potential objects. These methods are relatively slow and may not always provide accurate or optimal proposals. R-CNN is also computationally expensive during both training and inference. It involves multiple stages, which makes it slow and resource-intensive.

3.2.3. Fast R-CNN

Fast R-CNN is an improved version of R-CNN that addresses some of its limitations. Spatial pyramid pooling networks (SPPnets) were proposed to speed up R-CNN by sharing computation. It computes a convolutional feature map for the entire input image and then classifies each object proposal.

The fast R-CNN model is visualized in 3.10. The Fast R-CNN architecture takes an input image and multiple regions of interest (RoIs) as input. It utilizes a fully convolutional network to process the entire image, which involves several convolutional and max pooling layers, resulting in a convolutional feature map. For each object proposal, a RoI pooling layer is employed to extract a fixed-length feature vector from the convolutional feature map. This feature vector is then passed through a sequence of fully connected (FC) layers. The FC layers branch into two output layers: The first output layer produces softmax probability estimates for K object classes, including a ”background” class that represents non-object regions. The second output layer generates four real-valued numbers for each of the K object classes, representing refined bounding-box positions.

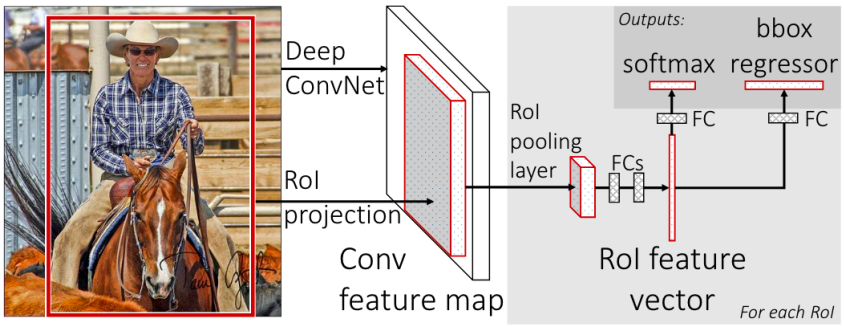


Figure 3.9: Fast R-CNN architecture [10, Fig. 1]

3.2.4. Faster R-CNN

Faster R-CNN, proposed in 2015, revolutionized object detection by introducing a two-stage framework that combines region proposal generation and object classification. It is the most widely adopted two-stage architecture. The primary motivation behind Faster R-CNN was to alleviate the shortcomings of earlier approaches that relied on time-consuming external region proposal methods, such as Selective Search. It consists of a region proposal network (RPN) and a subsequent object detection network. The RPN shares convolutional layers with the detection network, allowing for efficient computation and reuse. This design ensures end-to-end training and enables the network to learn powerful representations for both region proposal and object classification. The network is presented in Figure 3.10

However, the two-stage approaches mentioned above suffer from increased computational complexity due to the need for region proposal techniques. This limitation makes them less suitable for real-time applications where efficiency is a critical factor.

One-stage methods treat object detection as a regression or classification problem and use a unified framework to achieve final results directly. One-stage methods include the family of You Only Look Once (YOLO) [4], SSD [12], etc.

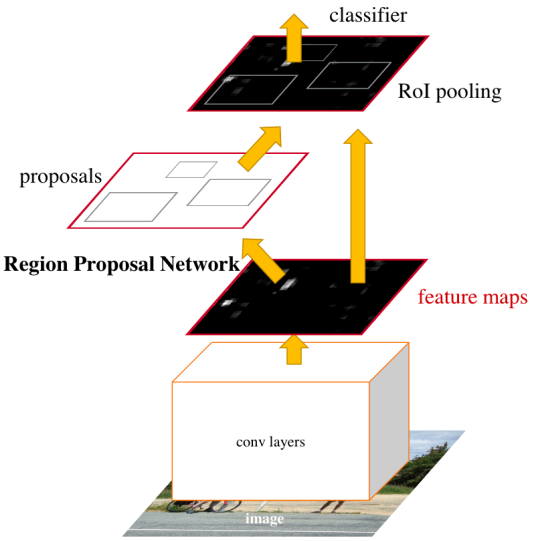
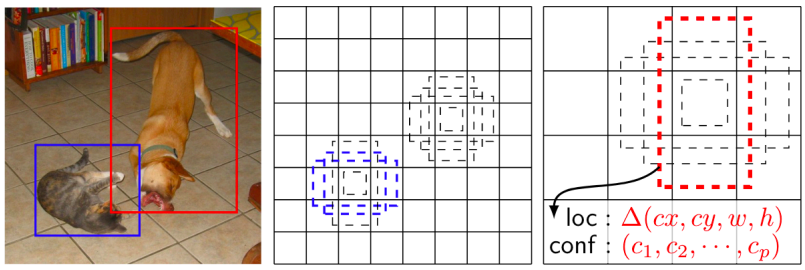


Figure 3.10: Faster R-CNN model structure [2, Fig. 2]

3.2.5. SSD

Unlike two-stage methods, SSD is a one-stage detector that performs both object localization and classification in a single pass through the network.

The key idea behind SSD is the use of multiple convolutional feature maps at different scales to detect objects of various sizes. These feature maps are obtained from different layers of a base network, such as VGG or ResNet. Each feature map is associated with a set of default bounding boxes of different aspect ratios and scales, which act as anchors for object detection. SSD framework is shown in 3.11. It is relatively simple compared to methods that require object proposals because it completely eliminates proposal generation. Instead, all the necessary computations for object detection are encapsulated within a single network.



(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

Figure 3.11: SSD framework [12, Fig. 1]

3.2.6. RetinaNet

RetinaNet is an object detection model that addresses the problem of accurately detecting objects at different scales and handling the issue of class imbalance in training data.

One of the key features of RetinaNet is its focal loss, which effectively addresses the problem of class imbalance. Class imbalance occurs when the number of background (non-object) samples significantly outweighs the number of object samples in the training data. The focal loss assigns higher weights to hard examples (e.g., rare objects or misclassified samples) during training, thus focusing the model's attention on these challenging instances and improving overall performance.

Focal Loss adds a factor $(1-p_t)^\gamma$ to the standard cross entropy criterion. Setting $\gamma > 0$ reduces the relative loss for well-classified examples ($p_t > .5$), putting more focus on hard, misclassified examples.

RetinaNet employs a feature pyramid network (FPN) as its backbone architecture, which allows

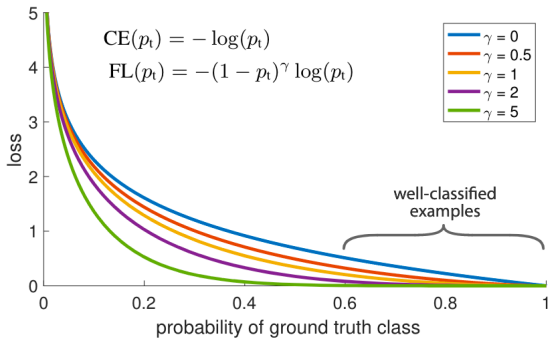


Figure 3.12: Focal Loss [3, Fig. 1]

the model to efficiently capture object information at multiple scales. The FPN combines features from different layers of a convolutional network to create a feature pyramid, enabling the detection of objects of various sizes. This multi-scale approach is particularly important for detecting small objects that might be missed by traditional single-scale methods. The network is shown in 3.13.

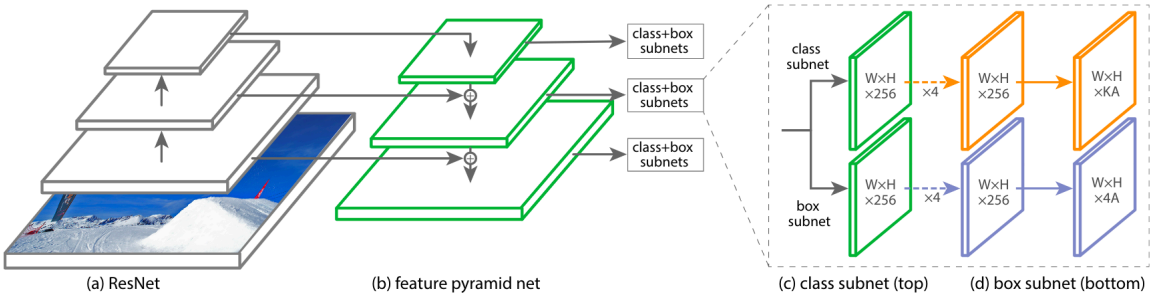


Figure 3.13: RetinaNet network uses a FPN backbone [3, Fig. 3]

3.2.7. YOLO

Redmon et al. [4] proposed a framework called YOLO. The YOLO model processes the entire image in a single feed-forward pass through a neural network and directly predicts the class probabilities and bounding box coordinates for a set of predefined anchor boxes. This approach is in contrast to traditional object detection systems that first generate region proposals and then classify them.

The basic idea of YOLO is displayed in Figure 3.14. The model divides the input image into a grid of cells, and each cell is responsible for predicting the bounding boxes and class probabilities for a set of predefined anchor boxes. Each bounding box consists of four parameters: x, y, width, and height. The class probabilities represent the likelihood of each anchor box containing a particular class.

The architecture of the YOLO model is displayed in Figure 3.15. The network has 24 convolutional layers and 2 fully connected layers. 1×1 convolutional layer is used to reduce the feature dimension from previous layers.

To make predictions, the YOLO model first processed the input image through a series of convolutional layers to extract features. The resulting feature map is then divided into a grid of cells. Each call predicts a fixed number of bounding boxes.

During training, the model learns to adjust the anchor boxes' parameters to better fit the ground-truth bounding boxes in the training data. The loss function used to train the model consists of a combination of localization loss (how well the predicted bounding boxes match the ground-truth boxes) and classification loss (how accurately the model predicts the object classes).

YOLO is designed to operate at high frame rates, making it suitable for real-time applications such as video analysis, robotics, and surveillance. It achieves this speed by using a single neural network for both object proposal and classification, eliminating the need for separate region proposal and object classification stages.

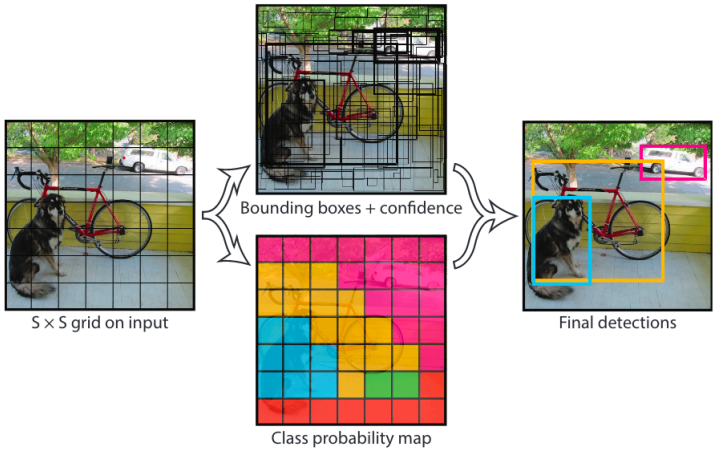


Figure 3.14: Main idea of YOLO [4, Fig. 2]

3.2.8. YOLOv5

YOLOv5 [13] is an object detection algorithm that builds upon the YOLO (You Only Look Once) architecture. It is one of the most commonly used models in the YOLO family of computer vision. It was developed as an evolution of the YOLOv4 model with the goal of improving speed, accuracy, and overall performance. YOLOv5 follows a one-stage object detection approach, which means that it directly predicts bounding boxes and class probabilities in a single pass. The architecture consists of a backbone network (CSPDarknet) followed by detection heads.

Here are some key aspects of YOLOv5:

- Anchor-based: YOLOv5 uses anchor boxes to help the detection of objects at various scales and aspect ratios. These anchor boxes serve as reference priors for predicting bounding boxes.
- Scaled-YOLO: YOLOv5 introduces ”Scaled-YOLO” which involves training the detector at multiple input sizes. By varying the input resolution during training, YOLOv5 is able to generalize better across different object scales.
- Different Model Sizes: YOLOv5 provides different model sizes, such as YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x, with increasing depth and capacity. The larger models tend to offer better accuracy but require more computational resources.
- Improved Performance: YOLOv5 incorporates several enhancements to improve performance. These include the use of advanced data augmentation techniques, the introduction of PANet (Path Aggregation Network) for feature fusion across different scales, and the utilization of anchor-based box predictors.
- Training and Inference: YOLOv5 can be trained on large-scale labeled datasets, such as COCO (Common Objects in Context), and the training process involves optimizing various loss functions to improve object localization and classification. Inference with YOLOv5 involves passing an image through the trained network, generating bounding box predictions, and applying post-processing steps to filter and refine the detections.
- Open-Source Implementation: YOLOv5 is open-source and provides a user-friendly code that allows researchers and developers to train and deploy the models for object detection tasks. The code includes pre-trained models, data augmentation techniques, evaluation metrics, and visualization tools.

Since its ease of use, high performance, and versatility, we decide to use it as a benchmark.

3.3. Imbalanced Dataset

3.3.1. Problem definition

An imbalanced dataset refers to a dataset in which the distribution of classes or samples across different categories is significantly unequal. In other words, some categories have more samples than others.

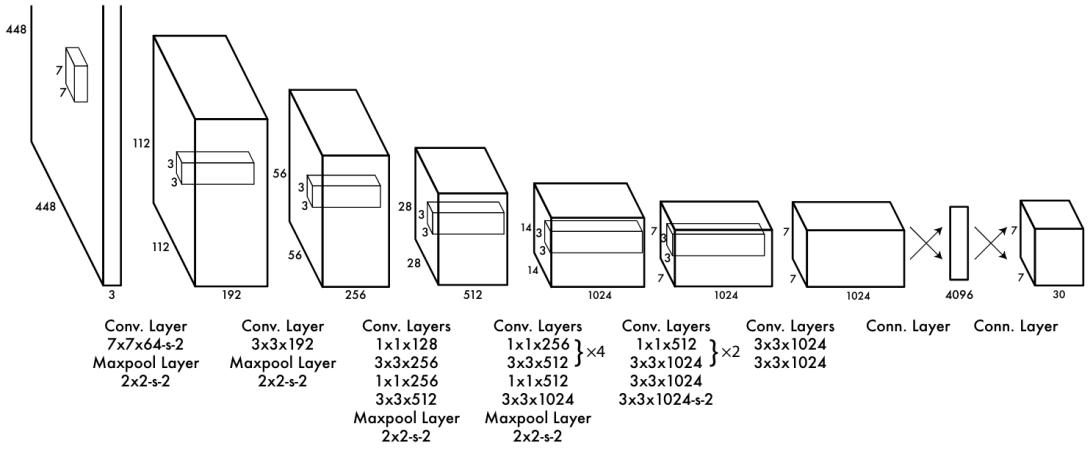


Figure 3.15: The architecture of YOLO model [4, Fig. 3]

This imbalance is commonly observed in various real-world domains, such as fraud detection, medical diagnosis, etc.

Due to the imbalanced representation of classes, models tend to be biased towards the majority class and may struggle to classify instances from the minority class accurately. This can result in poor overall performance for the minority class.

3.3.2. Approaches

Sampling Techniques

Sampling techniques involve modifying the class distribution in the dataset by either oversampling the minority class or undersampling the majority class.

Oversampling increases the number of instances in the minority class. Techniques like Random Oversampling, and SMOTE (Synthetic Minority Over-sampling Technique) [14] generate synthetic samples to balance the classes. Undersampling involves reducing the number of instances in the majority class.

Data Augmentation

Data augmentation techniques, such as rotation, scaling, flipping, or adding noise can help increase the diversity and quantity of data for underrepresented classes. This can help improve the model's ability to learn from and generalize to these classes.

Focal Loss

The focal loss [3] addresses the class imbalance problem by assigning higher weights to misclassified examples from the minority classes during training. The focal loss achieves this by introducing a modulating factor called the ”focusing parameter” that decreases the loss assigned to well-classified examples. This docusing parameter reduces the loss for easily classified examples, which are typically from the majority class, and increase the loss for hard examples, which are typically from the minority class. By giving more importance to the underrepresented classes, the model can focus on learning their distinctive features and improving their detection performance.

However, it is worth noting that focal loss is just one of several techniques avaiable to tackle the class imbalance problem. Its effectiveness can vary depending on the specific dataset and task.

3.4. Few Shot Object Detection

3.4.1. Problem Definition

Few-shot object detection refers to the task of detecting objects in an image with limited labeled training examples per object class. Traditional object detection methods typically require a large number of labeled examples for each object class to achieve good performance. However, in few-shot object

detection, the goal is to develop models that can effectively detect objects with only a small number of labeled examples per class, often as few as one or a few shots.

3.4.2. Approaches

Data Augmentation

Data augmentation is a technique used in machine learning and computer vision to artificially increase the diversity and size of a training dataset by applying various transformations and modifications to the existing data. The goal of data augmentation is to improve the generalization and robustness of the trained models. It is a commonly used solution to few-shot problems. Data augmentation techniques used in this thesis are visualized in 3.16.

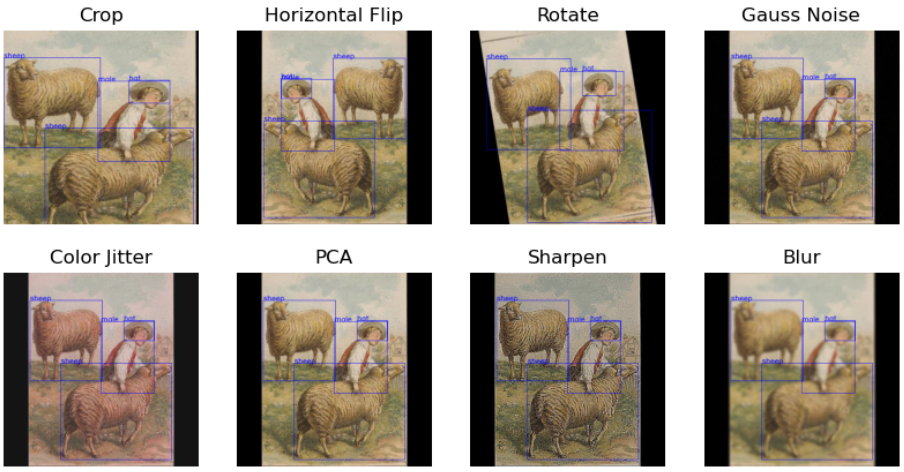


Figure 3.16: Eight augmentation techniques

Data augmentation techniques can be broadly categorized into several categories based on the types of transformations applied to the original data. Here are some common categories of data augmentation:

- Geometric Transformations: These techniques involve geometric modifications to the data, such as rotation, translation, scaling, flipping, and cropping. These transformations help the model to learn invariance to different orientations, positions, and sizes of objects.
- Color Transformations: These techniques alter the color properties of the data, such as adjusting brightness, contrast, saturation, hue, or applying color filters. Color augmentation can help the model become more robust to variations in lighting conditions, color shifts, and contrast levels.
- Noise Injection: Adding different types of noise to the data can improve the model's ability to handle noisy or distorted inputs. Examples include Gaussian noise, random pixel value noise, or dropout, where random pixels or regions are set to zero.
- Occlusion and Cutout: Introducing occlusions or cutout regions in the data can force the model to focus on relevant features and improve its robustness to occluded objects or missing regions.
- Style Transfer: Style transfer techniques can be used to change the visual style or appearance of the data, such as applying artistic filters or transferring the style of one image to another. This can help the model learn to recognize objects in different visual styles.

Transfer Learning

Transfer learning is a machine learning technique where knowledge gained from training a model on one task or dataset is transferred and applied to a different task or dataset. Normally, the dataset that has been pre-trained is abundant, while the new dataset suffers from scarcity. It utilizes the learned features from a pre-trained model to improve the performance and efficiency of a new task.

In the context of object detection, transfer learning can be applied by utilizing pre-trained models that have been trained on large-scale datasets such as ImageNet [15]. These pre-trained models have learned generic visual features that are useful for various computer vision tasks. Transfer learning in

object detection allows us to benefit from the knowledge and representations learned from large-scale datasets, even when the target dataset is relatively small. It helps in overcoming the limitations of insufficient labeled data and reduces the training time and computational resources required to achieve good performance.

For instance, Wang et al. [16] adopted a two-stage fine-tuning approach for few-shot object detection. It is simple but also effective. The process is presented in 3.17.

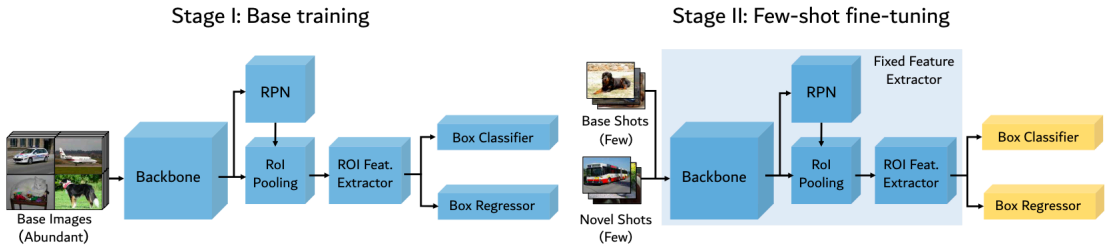


Figure 3.17: Illustration of two-stage fine-tuning approach [16, Fig. 2]

While transfer learning has been extensively used in few-shot classification (FSC) tasks, its application in few-shot object detection (FSOD) is more challenging. First, applying the conventional transfer learning strategy of initializing deep detectors from pre-trained deep classifiers is not suitable when dealing with limited target detection sets. The reason is that fine-tuning on such small target sets is often hard to eliminate the task difference between detection and classification. Second, deep object detectors are more prone to overfitting compared to deep classifiers as they need to learn object-specific representations for both localization and classification [17].

Distance Metric Learning

Distance metric learning aims to learn a suitable distance metric that can effectively measure the similarity or dissimilarity between feature representations of objects. By leveraging this learned distance metric, it becomes possible to compare and classify novel objects based on their feature similarity to the few-shot training examples.

One popular approach in distance metric learning is to use Siamese networks [18] or triplet networks [19]. These networks learn embeddings for object instances such that instances from the same class are closer together in the embedding space, while instances from different classes are pushed further apart. This way, during inference, the distance between a novel object instance and the few-shot examples can be used to determine its class.

3.5. Datasets

3.5.1. Children Books

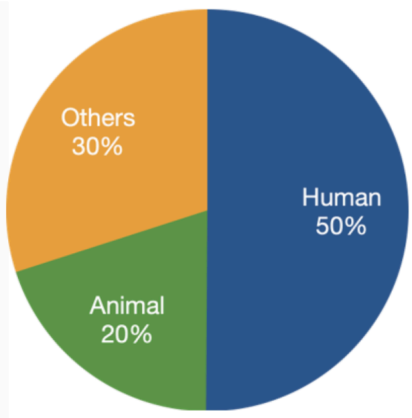


Figure 3.18: Pie Chart of the Original Dataset [5]

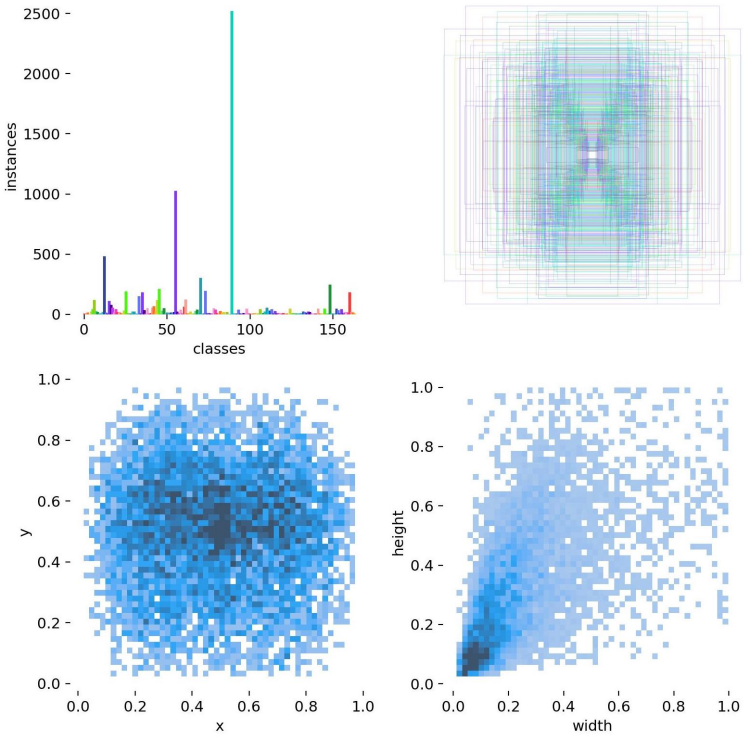


Figure 3.19: Label information of the Original Dataset

The original dataset is the Ot & Sien dataset from `https://lab.kb.nl/dataset/ot-sien-dataset`. It has 1512 illustration images with 7210 objects. (4.8 per image), annotated with object class and bounding boxes. It has 264 classes and is super imbalanced. The distribution of categories is visualized in 3.18. Our dataset is very imbalanced, 3.18 captures the situation. Human categories, including ”male” and ”female”, make up around 50% of all objects.

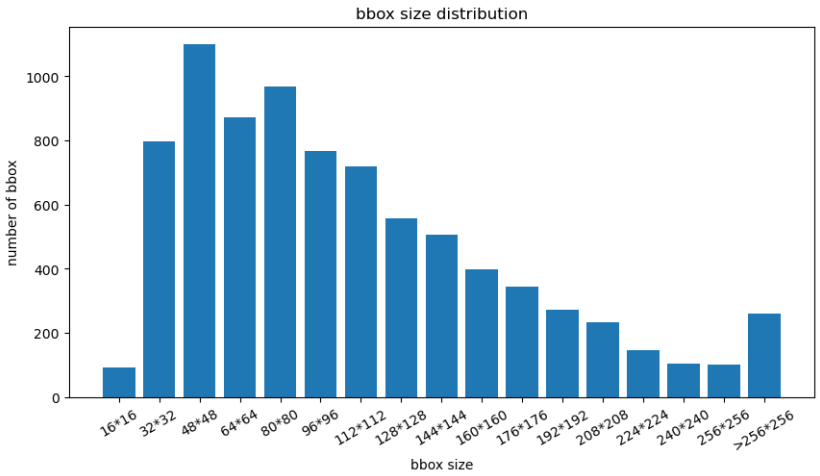


Figure 3.20: The distribution of the size of bounding boxes

After looking into the dataset, many mislabeling and wrong classes are identified. After correction, the properties of the dataset can be summarized as:

- The dataset consists of illustrations rather than standard photos.
- 1452 images with 8241 objects (5.7 per image) are annotated including the category and bounding boxes.

- All images are resized to 416×416 with black fitting edges to adapt to the training procedure.
- The dataset follows a natural long-tail property, with some object categories being rare.
- The dataset has imbalanced categories.

The information on bounding boxes is presented in 3.19. The first figure in 3.19 displays the instances-per-class distribution. This distribution high- lights the inherent imbalance and low-shot nature of the dataset, with certain categories having a significantly larger number of instances compared to others.

The second figure shows the shape and location of all bounding boxes in our dataset and the third figure shows the distribution of the center of all bounding boxes. The visualization demonstrates the spatial distribution of objects and illustrates that many bounding boxes are centered around the middle of the image.

Lastly, the size distribution of the bounding boxes is depicted in the last figure of Figure 2. This visualization illustrates that the dataset contains a significant number of small objects, as indicated by the higher density of data points in the lower region of the size distribution plot.

The detailed distribution of the size of bounding boxes is visualized in 3.20. We can see that the dataset contains a higher number of small objects compared to larger ones. This observation underscores the importance of effectively detecting and classifying smaller objects within the dataset.

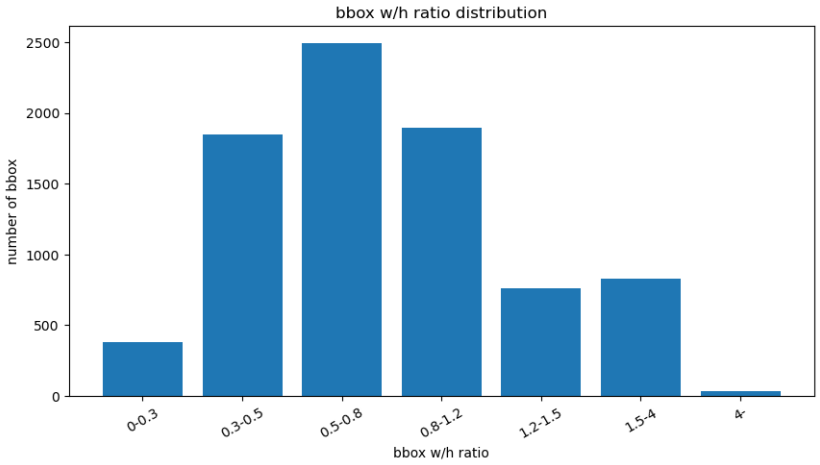


Figure 3.21: The distribution of width/height ratio of bounding boxes

3.5.2. Commonly Used Datasets for Object Detection

PASCAL VOC 2012 [20] is a mid-scale object detection dataset that shares the same 20 categories as Pascal VOC 2007. PASCAL VOC has three splits: training, validation, and test, containing 5717, 5823, and 10991 images, respectively. The main purpose of this dataset is to recognize objects from a variety of visual object classes in realistic settings. It is a supervised learning problem with a training set of labeled images provided. It contains twenty categories:

- Person: person
- Animal: bird, cat, cow, dog, horse, sheep
- Vehicle: aeroplane, bicycle, boat, bus, car, motorbike, train
- Indoor: bottle, chair, dining table, potted plant, sofa, tv/monitor

MicroSoft COCO [6] is a large-scale object detection, segmentation, and captioning dataset with 91 categories. It collects images of complex everyday scenes containing common objects in their natural context. The number of instances per category for all 91 categories is shown in 3.22.

LVIS (Large Vocabulary Instance Segmentation) [21], is an extensive dataset aimed at advancing the field of instance segmentation. The dataset is designed to be comprehensive, comprising approximately 164,000 images, over 1000 categories, and over 2 million high-quality instance segmentation masks. It specifically focuses on categories with limited training samples, encompassing a long-tail

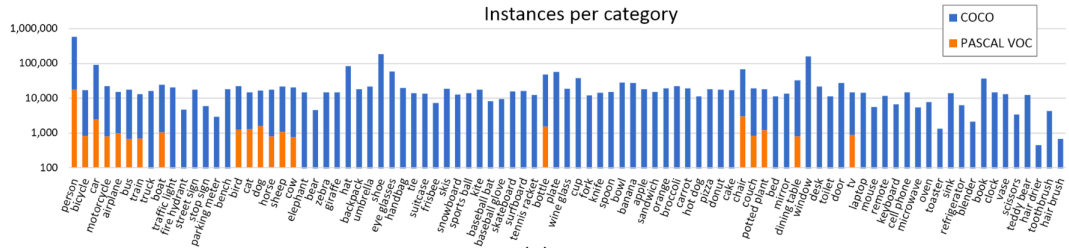


Figure 3.22: categories distribution of MS COCO [6, Fig. 5 (a)]

distribution. The dataset provides a benchmark challenge, serving as a platform for the development of novel object detection, segmentation, and few-shot learning algorithms.

Few Shot Object Detection (FSOD) [22] is a highly diverse dataset created specifically for few-shot object detection. It is designed to access a model's generality on new categories. The training set and test set have no overlapping categories, having 800 and 200 categories each. The training set is constructed from the MS COCO dataset and the ImageNet dataset. Overall, the dataset contains 1000 categories, of which 531 categories come from ImageNet and 469 from the Open Image dataset.

The information and statistics of these datasets are presented in 3.1 and 3.2.

Dataset	Classes	Object Per Image	Image Size	Started Year
Pascal Voc 2012	20	2.4	470 $\times$ 380	2005
MS COCO	91	7.3	640 $\times$ 480	2009
LVIS	1000+	11.2	-	2019
FSOD	1000	2.82	-	2020
Children Books	164	5.8	416 $\times$416	2023

Table 3.1: Characteristics of Considered Datasets

Dataset	Numebr of Images			Number of Annotations	
	Train	Val	Test	Train	Val
Pascal Voc 2012	5,717	5,823	10,991	13,609	13,841
MS COCO	82,783	40,504	81,434	604,907	291,875
LVIS	100,170	19,809	19,822	1,270,141	244,707
FSOD	52,350	14,152	–	147,489	35,102

Table 3.2: Statistics of the commonly used Datasets in FSOD

References

[1] Aakash K Shetty et al. “A review: Object detection models”. In: *2021 6th International Conference for Convergence in Technology (I2CT)*. IEEE. 2021, pp. 1–8.

[2] Ross Girshick. “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[3] Tsung-Yi Lin et al. “Focal loss for dense object detection”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.

[4] Joseph Redmon et al. “You only look once: Unified, real-time object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[5] S. Khademi, S. Veldhoen, and L. Wilms. *Ot & Sien dataset*. `https://lab.kb.nl/dataset/ot-sien-dataset`. KB Lab: The Hague. 2021.

[6] Tsung-Yi Lin et al. “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.

[7] Keiron O’Shea and Ryan Nash. “An introduction to convolutional neural networks”. In: *arXiv preprint arXiv:1511.08458* (2015).

[8] Xiongwei Wu, Doyen Sahoo, and Steven CH Hoi. “Recent advances in deep learning for object detection”. In: *Neurocomputing* 396 (2020), pp. 39–64.

[9] Ross Girshick et al. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[10] Ross Girshick. “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[11] Kaiming He et al. “Mask r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969.

[12] Wei Liu et al. “Ssd: Single shot multibox detector”. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer. 2016, pp. 21–37.

[13] Glenn Jocher et. al. *ultralytics/yolov5: v6.0 - YOLOv5n ’Nano’ models, Roboflow integration, TensorFlow export, OpenCV DNN support*. Version v6.0. Oct. 2021. DOI: `10.5281/zenodo.5563715`. URL: `https://doi.org/10.5281/zenodo.5563715`.

[14] Nitesh V Chawla et al. “SMOTE: synthetic minority over-sampling technique”. In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[15] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[16] Xin Wang et al. “Frustratingly simple few-shot object detection”. In: *arXiv preprint arXiv:2003.06957* (2020).

[17] Hao Chen et al. “Lstd: A low-shot transfer detector for object detection”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.

[18] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. “Siamese neural networks for one-shot image recognition”. In: *ICML deep learning workshop*. Vol. 2. 1. Lille. 2015.

[19] Elad Hoffer and Nir Ailon. “Deep metric learning using triplet network”. In: *Similarity-Based Pattern Recognition: Third International Workshop, SIMBAD 2015, Copenhagen, Denmark, October 12-14, 2015. Proceedings 3*. Springer. 2015, pp. 84–92.

[20] Mark Everingham et al. “The pascal visual object classes (voc) challenge”. In: *International journal of computer vision* 88 (2010), pp. 303–338.

[21] Agrim Gupta, Piotr Dollar, and Ross Girshick. “Lvis: A dataset for large vocabulary instance segmentation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 5356–5364.

[22] Qi Fan et al. “Few-Shot Object Detection with Attention-RPN and Multi-Relation Detector”. In: *CVPR*. 2020.