Continual Object Detection with Deep Neural Networks

Angelo Garangau Menezes

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

Angelo Garangau Menezes

Continual Object Detection with Deep Neural Networks

Doctoral dissertation submitted to the Instituto de Ciências Matemáticas e de Computação, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho

USP – São Carlos
November 2023

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

```
         Menezes, Angelo Garangau
M543c       Continual Object Detection with Deep Neural
         Networks / Angelo Garangau Menezes; orientador
         André Carlos Ponce de Leon Ferreira  de Carvalho. --
          São Carlos, 2023.
            113 p.


            Tese (Doutorado - Programa de Pós-Graduação em
         Ciências de Computação e Matemática Computacional) --
          Instituto de Ciências Matemáticas e de Computação,
         Universidade de São Paulo, 2023.


            1. Object Detection. 2. Continual Learning. 3.
         Incremental Object Detection. I. de Carvalho, André
         Carlos Ponce de Leon Ferreira , orient. II. Título.
```

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Angelo Garangau Menezes

Aprendizado Contínuo de Objetos com Redes Neurais Profundas

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho

USP – São Carlos
Novembro de 2023

ACKNOWLEDGEMENTS

To all the good teachers and professors I had in my life who made me believe in science and that serving others makes a life worth living.

To all the bad teachers and professors who somehow fueled me to get a Ph.D. and try to make a positive difference in other people's lives, as they should have done, but didn't.

To my lovely wife Priscila, my son Vicente, and my family, without whom I would not stand a chance in this Ph.D.

To my supervisor, prof. André Carvalho, who granted me innumerous learning opportunities since before my Ph.D. through his mentorship and invaluable conversations.

To Vincenzo Lomonaco and Julio Hurtado for all the insights in continual learning for research and life.

To all the friends I made during my 11 years in Academia who made me a better person and researcher.

To God, for allowing us to live in the great simulation we call life.

The author would like to thank Taesa for the financial support through the ANEEL R&D program, process PD-07130-0059/2020.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

RESUMO

MENEZES, A. G. **Aprendizado Contínuo de Objetos com Redes Neurais Profundas**. 2023. 113 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

O rápido desenvolvimento tecnológico nas últimas décadas aumentou significativamente a quantidade de dados disponíveis no mundo. Naturalmente, modelos que escalam com o tamanho dos dados disponíveis, como as redes neurais profundas, tornaram-se a principal estratégia para vários campos de pesquisa com abundância de dados, como por exemplo visão computacional e processamento de linguagem natural. Com a grande disponibilidade de dados, a pesquisa sobre modelos de aprendizado que podem se adaptar de forma incremental a fluxos contínuos de dados tem sido incentivada. Dessa forma, a área de Aprendizado Contínuo de modelos se apresenta como o campo que propõe o estudo sobre a capacidade de aprender tarefas consecutivas sem perder desempenho nas tarefas previamente treinadas. Para a área de visão computacional, os pesquisadores têm concentrado seus esforços principalmente em tarefas de classificação incremental, mas a detecção contínua de objetos também merece atenção devido à sua vasta gama de aplicações em robótica e veículos autônomos. O cenário de detecção incremental é ainda mais complexo que a simples classificação devido à ocorrência de instâncias de classes desconhecidas mas que podem aparecer em tarefas subsequentes como uma nova classe a ser aprendida, resultando em anotações ausentes e conflitos com o rótulo de *background*. Uma vez que se apresenta em seus estágios iniciais, a pesquisa em detecção contínua de objetos ainda oferece várias oportunidades e carece de convenções metodológicas. Desta maneira, esta tese de doutorado busca investigar esse campo mais detalhadamente e identificar possíveis vínculos com áreas relacionadas, como aprendizado contínuo geral e a poda de redes neurais. Especificamente, propusemos a primeira revisão sistemática sobre o tópico, desenvolvemos duas métricas para melhorar a análise de desempenho em cenários de detecção incremental, investigamos qual método de seleção de exemplares funciona melhor para estratégias de detecção contínua de objetos baseadas em *replay* e exploramos como identificar e penalizar parâmetros importantes de tarefas que possuam treinamento contínuo. Para validar nossas propostas e hipóteses, conduzimos experimentos e relatamos resultados comparáveis ao estado da arte atual em benchmarks populares de detecção (ex: PASCAL VOC) adaptados à configuração incremental, bem como em conjuntos de dados e aplicações do mundo real. As contribuições apresentadas nesta tese também foram colocados em prática em duas aplicações. Primeiramente, elas foram testados no 3rd CLVISION Challenge, onde alcançaram a 3rd posição na trilha de detecção contínua de instâncias. Em segundo lugar, foram aplicadas na inspeção aérea contínua de torres de transmissão da TAESA, maior empresa brasileira de transmissão de energia elétrica, para melhora de suas *pipelines* de inspeção automatizada.

Palavras-chave: Detecção de Objetos, Aprendizado Contínuo, Detecção de Objetos Incremental, *Replay*, Mineração de Parâmetros.

ABSTRACT

MENEZES, A. G. **Continual Object Detection with Deep Neural Networks**. 2023. 113 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

The rapid technological development in the past decades has significantly increased the amount of available data in the world. Naturally, models that scale with the size of the available data, such as Deep Neural Networks, have become the primary strategy for several research fields with abundant data (e.g., computer vision and natural language processing). With this large data availability, research on learning models that can adapt incrementally to continual streams of data has been encouraged. In this way, the field of Continual Learning proposes to study the ability to learn consecutive tasks without losing performance on the previously trained ones. In computer vision, researchers have mainly focused their efforts on incremental classification tasks, but continual object detection also deserves attention due to its vast range of applications in robotics and autonomous vehicles. In fact, this scenario is even more complex than conventional classification, given the occurrence of instances of classes that are unknown at the time but can appear in subsequent tasks as a new class to be learned, resulting in missing annotations and conflicts with the background label. Since this field is in its early stages, research in continual object detection still offers several opportunities and lacks methodology conventions. This Ph.D. thesis investigates the field more thoroughly and identifies possible links with related areas such as general continual learning and neural network pruning. Specifically, we proposed the first systematic review on the topic, developed two metrics for improving the analysis of performance in incremental detection scenarios, investigated which exemplar selection method works best for replay-based continual detection strategies, and explored different ways to identify and penalize important task parameters across sequential updates. To validate our proposals and claims, we conducted experiments and reported results comparable to the current state-of-the-art in popular detection benchmarks (i.e., PASCAL VOC) adapted to the incremental setting, as well as in real-world datasets and applications. The findings presented in this thesis were also put into practice in two applications. Firstly, they were tested in the 3rd CLVISION Challenge, where we were able to achieve the 3rd place in the continual instance detection track. Secondly, they were applied to the continual aerial inspection of transmission towers at TAESA, the largest Brazilian electric power transmission company, to improve the automation of their inspection pipeline.

Keywords: Object Detection, Continual Learning, Continual Object Detection, Replay, Parameter Mining.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

mAP	mean average precision
ACC	Average Accuracy
AGI	artificial general intelligence
AP	average precision
AR	average recall
AUC	area under the curve
BWT	Backward Transfer
CF	catastrophic forgetting
CIOD	Class-Incremental Object Detection
CL	Continual Learning
CNNs	Convolutional Neural Networks
COD	Continual Object Detection
DIOD	Domain-Incremental Object Detection
DNNs	Deep Neural Networks
EAP	Experience Average Precision
EWC	Elastic Weight Consolidation
FPN	Feature Pyramid Networks
FWT	Forward Transfer
GEM	Gradient Episodic Memory
ILOD	Knowledge distillation strategy for COD proposed by Shmelkov, Schmid and Alahari (2017)
IOU	intersection over union
KL	Kullback Leibler
LwF	Learning Without Forgetting
MLP	Multilayer Perceptron
MMN	Parameter-isolation strategy for COD proposed by Li *et al.* (2018)
NMS	Non-Maximum Supression
ResNet	residual network
RILOD	Knowledge distillation strategy for COD proposed by Li *et al.* (2019)
RoI	Region of Interest
RPN	Region Proposal Network

SSD	Single Shot Multibox Detector
UAVs	Unmanned Aerial Vehicles
YOLO	You Only Look Once

CONTENTS

CHAPTER 1

INTRODUCTION

This introductory chapter provides a brief contextualization of the research area where this Ph.D. thesis fits in. We discuss some aspects that corroborate our choice for tackling continual learning problems, especially for the continual object detection task, and demonstrate the relevance of the research theme. Finally, we introduce our main research questions, hypotheses, and final objectives.

1.1 Ph.D. Thesis Context

The fast technological development in the past decades has made the amount of available data increase at significant rates. In fact, the recurrent technical jargon that has been on the news to characterize this fact is the term *Big Data* (WALKER, 2014), which concerns, as the name implies, the manipulation of vast amounts of information. Even so, the degree of change present in this endless flow of data is only a small representation of how the world is constantly changing and evolving. To deal with that, our brains have developed several mechanisms to cope with the necessary adaptation to different experiences (GROSSBERG, 2012). This analogy has not only guided scientists during their early exploration of the neural principles of learning in the brain but also inspired the computational modeling of the learning and forgetting functions in artificial neurons (RATCLIFF, 1990; HASSABIS *et al.*, 2017).

Naturally, with computing power also becoming more available for the academic "masses", robust computational models that make use of large amounts of data began to exhibit unprecedented results in fields that only humans used to do well, such as vision and natural language understanding (LECUN; BENGIO; HINTON, 2015). Most of this success comes from the application of deep neural networks, which are models that, although having most of their theory dating back to the 80s and early 90s (LECUN *et al.*, 1988; LECUN; BENGIO *et al.*, 1995), have only risen to be the "go to" technique for machine learning after some groundbreaking results in 2012 (KRIZHEVSKY; SUTSKEVER; HINTON, 2012).

Deep Neural Networks (DNNs) are computationally distributed models able to learn representations from raw data through a structure of hierarchical layers, similar to how the brain handles new information. However, they are a powerful solution only when being used with data that is carefully shuffled, balanced, and standardized (HADSELL *et al.*, 2020). As real-world data may come in large streams and vary considerably from what was available during the initial training, some necessary assumptions for DNNs might not be met. In this case, they can fail entirely or suffer from a fast decay in performance for early learned tasks when trained sequentially, which is commonly described as catastrophic forgetting (CF) or catastrophic interference (ROBINS, 1995).

These circumstances have influenced the introduction of the Continual Learning (CL) field, in which techniques are mainly refined to deal with different data-dynamic scenarios. Although the interest in this area has grown notably since 2016 (PARISI *et al.*, 2019), over the years, several names have been used to refer to the search for models that continually adapt. Some of them are "incremental learning", "lifelong learning" and "never-ending learning". Yet, the recent desiderata assigned to CL models have become even broader and involve not only the forgetting aspect but also the scalability, computational efficiency, and fast adaptability features (DÍAZ-RODRÍGUEZ *et al.*, 2018). For the context of this thesis, we considered the current standard expression (a.k.a. continual learning) to frame all previously related nomenclature.

Several applications that deal with streams of images can benefit from having models that can naturally deal with changing and incremental contexts, such as autonomous cars, UAVs, and house robots (SHAHEEN *et al.*, 2021). Applications of CL for object detection are the main focus of this Ph.D., which will be discussed more deeply in the following sections.

1.2 Motivation

Within the context of computer vision, the search for strategies able to deal with the modeling of a dynamic world is not new (ROSS *et al.*, 2008). Notwithstanding, most of the current solutions for CL consider the classification task as its main conundrum. In this way, the task of continual object detection, which involves both learning continually to localize and classify object samples, is not yet well explored, having its foundational work dating back to 2017 (SHMELKOV; SCHMID; ALAHARI, 2017).

Continual Object Detection (COD) is a more complex task than conventional classification since the predictive model needs to deal with situations where new objects, that were unknown previously, appeared in the previous training data but were not labeled and therefore considered as "background". This issue affects the notion of "objectness" of the model and may interfere with its performance towards either favoring the detection of only previously known objects or exclusively the new ones. This tradeoff is also in part due to the natural "tug-of-war" effect that each task creates on the model parameters during training (HADSELL *et al.*, 2020).

Each task greedily tries to bring the model to the local minima, where the weights will be optimum for giving the best results for the current task. Hence, when processing tasks sequentially, each task "catastrophically" interferes with the model results for the others, causing the so-called "forgetting". An illustration of this process is shown in Figure 1. When the model is optimized with the losses of several tasks not sequentially but simultaneously, assuming that all task data is available, the learning process can be treated as a "multitask learning" situation, as depicted in the last image.

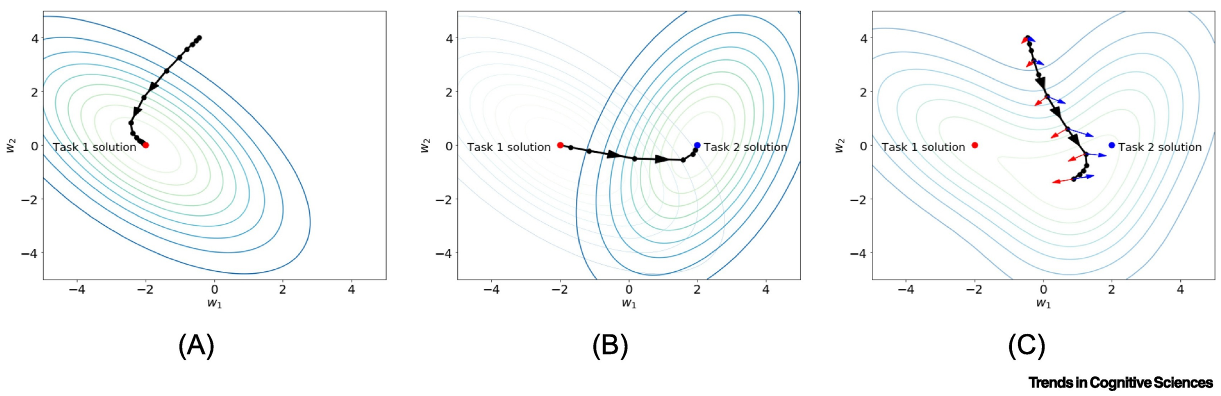


Figure 1 – An illustration of the tug-of-war that results in catastrophic forgetting. (HADSELL *et al.*, 2020).

Besides alleviating CF, it is usually expected that CL models present other important characteristics such as forward and backward transfer, fast adaptation, and computational efficiency (DÍAZ-RODRÍGUEZ *et al.*, 2018). Moreover, in addition to being an interesting research challenge, solutions to COD are relevant from an industry perspective (SHAHEEN *et al.*, 2021). A few of the main advantages of implementing robust COD solutions are:

- **Computational and Energy Efficiency**: Since the models do not have to be retrained from scratch each time, energy and computational time can be saved, which favors several applications with such constraints (e.g., aerial robotics).
- **Scalability**: Considering that often all training data is not available from time t to time $t+1$, CL usually provides solutions that focus on retaining only what is essential from each previous training experience, which favors scalability to process large chunks of data.
- **Privacy and Edge Computing**: Once a model is deployed, the new stream of data may only be used for updating it locally, which benefits edge computing and many other privacy concerns.

Research in CL for classification has been the main target of researchers over the years, which resulted in the investigation of several criteria that favors the adaptation of DNNs to the class-incremental learning scenario. The proposed solutions include changes in the architecture to make it wider (MIRZADEH *et al.*, 2021), the use of different learning paradigms (GALLARDO; HAYES; KANAN, 2021; BEAULIEU *et al.*, 2020) or training regimes (MIRZADEH *et al.*, 2020), and "plugins" that hinder forgetting (e.g., replay, regularization, and dynamic architectures) (DELANGE *et al.*, 2021). Nevertheless, for the continual object detection task, most of the existing solutions are still being proposed on variations of the same regularization techniques (PENG; ZHAO; LOVELL, 2020; HAO *et al.*, 2019; CHEN; YU; CHEN, 2019). In this way, we have realized that there is still a gap where one can investigate and explore strategies for COD taking advantage of what has and has not worked for classification in general CL and other fields, such as the use of different replay strategies and the insights from the neural network pruning literature. Also, considering the application aspect, there is still a lack of practical evaluations and reports of COD models being implemented in the real world.

1.3 Hypotheses and Objectives

Given the aforementioned challenges and opportunities, in this Ph.D. research, we formulate the following research questions:

Main Research Question 1: *Are standard CL metrics enough to represent the gains and losses in class-incremental object detection ?*

Main Research Question 2: *Is random replay the most suitable exemplar selection strategy for replay-based solutions in class-incremental object detection ?*

Main Research Question 3: *Is parameter mining and freezing a strong baseline for continual object detection ?*

By analyzing techniques used for CL in general and performing a systematic review of the current literature that involves class-incremental object detection, we recognize the potential that different architectures and learning paradigms can bring to our application context.

Given these considerations, our main hypotheses for this Ph.D. project are:

Hypothesis 1. *Metrics specifically tailored to highlight changes in the stability-plasticity of a model are more suited to class-incremental object detection than standard CL metrics.*

Hypothesis 2. *Class-balanced replay buffers are more effective for class-incremental object detection than using random buffers.*

Hypothesis 3. *The use of a well-selected parameter mining and freezing strategy can enable deep neural network models to continually learn how to detect new objects while avoiding forgetting old ones.*

To assess our hypotheses, we delineated a set of **objectives** to pursue:

- Review the state-of-the-art for continual detection and their research trends.
- Investigate the use of different replay techniques specially tailored for class-incremental object detection.
- Evaluate the impact of parameter mining and freezing for continual object detection.
- Assess the challenges of applying continual object detection to real-world constrained scenarios.

By fulfilling our objectives as contributions, we aim to push forward the challenging field of COD.

1.4 Organization

The remaining topics of this document are structured as follows: we present in Chapter 2 an overview of the technical background needed for understanding the main contributions of this thesis. Following, in Chapter 3, we present a comprehensive review of the state-of-the-art (up to 2022) and related works that involve the fields of object detection and COD. Next, we investigate in Chapter 4 the application of several exemplar replay techniques to COD, followed by Chapter 5, which elaborates on the use of important parameter mining and freezing strategies for such a task. For Chapter 6, we explore two scenarios where COD can be applied in the real world. Finally, for Chapter 7, we discuss our final considerations, contributions, and future work.

CHAPTER

2

TECHNICAL BACKGROUND

This chapter aims to provide the necessary background for the discussions presented in this thesis. We start by giving short definitions for some of the used deep learning terminologies and then proceed to describe broader concepts related to continual learning and object detection.

2.1 Deep Learning

In the search for a learning model that is able to generalize to different types of problems, deep neural networks have been widely used by researchers due to their robustness and quality of results in tasks with unstructured data such as images, text, and audio (KRIZHEVSKY; SUTSKEVER; HINTON, 2012; DEVLIN *et al.*, 2018; OORD *et al.*, 2016). Their core idea is based on the generality of the simple artificial model of a neuron, called Perceptron, to perform input-output mapping by changing its weights and bias during the optimization of a loss function (LECUN; BENGIO; HINTON, 2015).

Single neurons can only map linear functions (HAYKIN, 2010). To surpass that, neurons usually use nonlinear activation functions and are stacked in interconnected hierarchical layers, as illustrated in Figure 2, that empower the model to learn complex mapping functions. These neural networks are called by definition Multilayer Perceptron (MLP), but they can also be referred to as Feed-Forward Networks or Fully Connected Networks. The intermediary layers between the input and output are called hidden layers and, when in great number, characterize the model as a deep neural network.

An artificial neural network with only one hidden layer is capable of representing any continuous function according to the universal approximation theorem (LECUN; BENGIO; HINTON, 2015). However, the theorem states neither the number of neurons nor the type of activation or optimization method. This encouraged the research of techniques that can be used along with DNNs and bring other inductive biases that facilitate learning the task at hand.

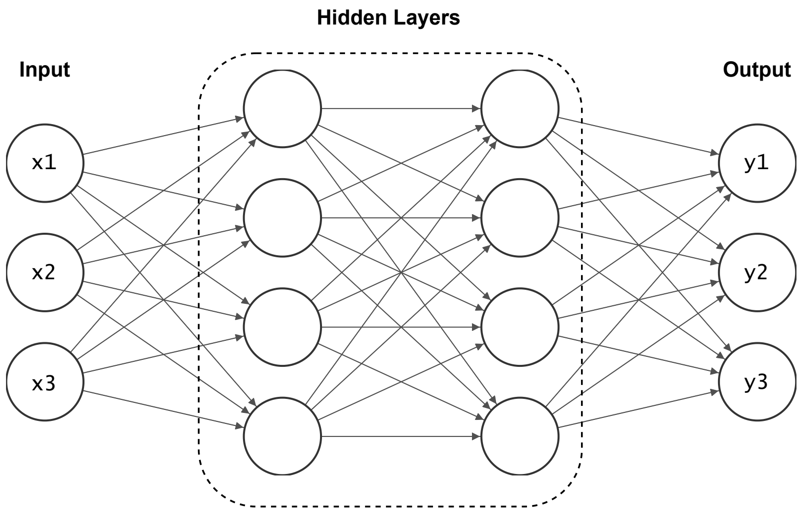


Figure 2 – MLP architecture with two hidden layers.

2.1.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) use the principle of sliding windows (i.e., kernels) and cross-correlation in their convolutional layers to extract discriminative feature maps from all image regions and pass them forward to be processed by neurons in the dense MLP layers (LECUN *et al.*, 1998). Unlike traditional computer vision pipelines, when using convolutional layers, it is unnecessary to specify the values within the kernel. Such values are treated as weights by the neural network and are optimized to precisely extract the patterns that best characterize each sample. Some advantages of using CNNs are:

- The patterns learned by kernels present translation invariance; that is, if the network has learned to recognize an object in the corner of the image, it will identify the object in any part of it.
- CNNs can learn hierarchical spatial patterns, which means that a first convolutional layer can learn to locate corners and edges in an image, while a second convolutional layer uses these discovered patterns as input to recognize more complex shapes, similarly to the human visual cortex. In this way, a CNN can have several stacked layers to improve its representation power.
- As the feature maps have a smaller dimension than the image itself, the number of parameters for training is reduced compared to those needed for purely dense MLP networks.
- The weights of the kernels are optimized along with the network itself, so it is not necessary to introduce human bias in the possible patterns to be detected.

Due to its strong representation power, CNNs are used effectively as the backbone of image classifiers, object detectors, and even audio generation by the use of visual spectrograms (LECUN; BENGIO; HINTON, 2015; OORD *et al.*, 2016). A typical CNN architecture used for computer vision tasks has convolutional layers with non-linear activations followed by pooling operators forming blocks, which can earn an identity of its own (e.g., VGG-Block) (SIMONYAN; ZISSERMAN, 2014). Additionally, blocks can be stacked and have top-down residual connections between them to facilitate gradient flow during optimization as presented in the residual network (ResNet) architecture (ALLEN-ZHU; LI, 2019). To perform non-linear operations with the learned presentations, fully connected layers are usually placed on top of the network, as illustrated by Figure 3.

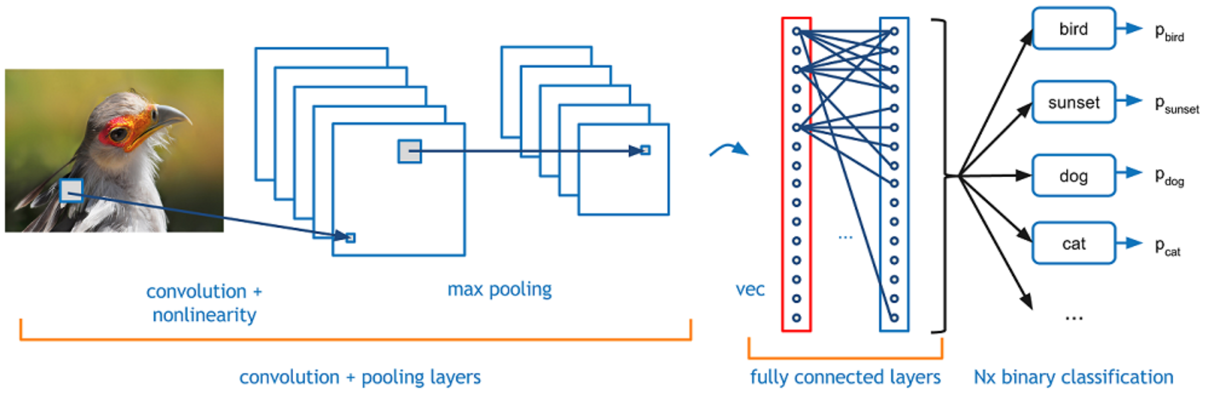


Figure 3 – CNN architecture exemplified (DESHPANDE, 2017)

2.1.2 Vision Transformers

Transformers are a type of DNN architecture that uses a self-attention mechanism for learning the relationships in elements of a sequence (VASWANI *et al.*, 2017). Their ability to scale according to the size of the data and model long-range relations has made it the standard architecture for text classification, machine translation, and question-answering tasks (KHAN *et al.*, 2021). Their core structure is based on an encoder-decoder setup as illustrated by Figure 4.

Recently, their success also reached the computer vision field with the adaptation of its pipeline to process images and videos successfully (CARION *et al.*, 2020; TOUVRON *et al.*, 2021). By processing an image as a sequence of positioned small patches, the model is able to encode global relationships of separate parts within an image that convolutions would struggle to identify (DOSOVITSKIY *et al.*, 2020). In contrast, since they make almost no assumptions about the structure of the data, the transformer might have difficulties when learning the inductive biases that CNNs have by default (e.g., translation equivariance). For having the best of both worlds, several solutions employ pre-trained CNNs as the backbone for large transformer encoder-decoder blocks (KHAN *et al.*, 2021).

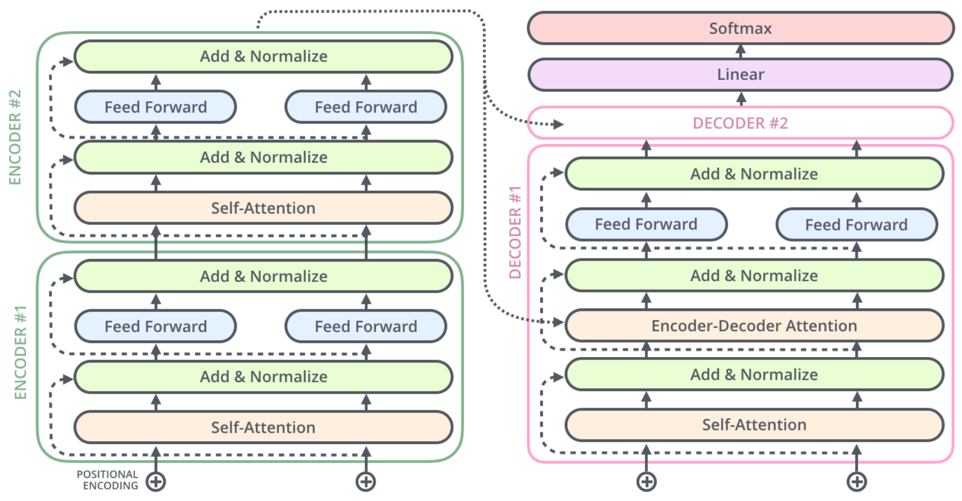


Figure 4 – A Transformer with two stacked encoders and decoders (ALAMMAR, 2018)

As transformers were designed to work with large unlabeled datasets, they are commonly pre-trained with pretext tasks in self-supervision setups, which helps to learn rich relationships between samples, and in this way, generalizable representations. Figure 5 shows an example of that when the self-supervised vision transformer proposed by (CARON *et al.*, 2021) demonstrated that object segmentation masks could be obtained from the last attention maps of the model even when not using any supervision to enforce that.

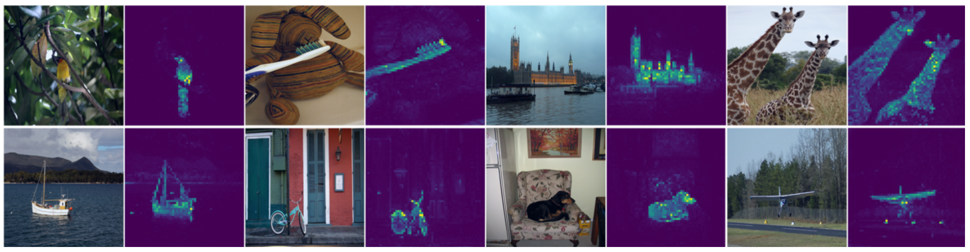


Figure 5 – DINO‘s attention maps (CARON *et al.*, 2021)

2.2 Continual Learning in Neural Networks

Continual learning, or lifelong learning, has been coined as the ability to learn consecutive tasks without forgetting how to perform on the previously trained ones (THRUN, 1995). Some researchers have pointed out over the years that research on this topic might lead to the development of an artificial general intelligence (AGI) (SILVER, 2011; CLUNE, 2019) since it is expected such behavior from intelligent agents.

For the scope of CL, we refer to *task* as a specific learning goal or problem that the model is attempting to solve. A CL model is sequentially trained to perform multiple tasks, such as recognizing digits, classifying cats and dogs, and detecting street signs. Each of these tasks would be considered a separate task for the purpose of CL as long as they are presented one at a time to the model. Unlike multi-task learning, when all the tasks are previously known, CL considers that new tasks are introduced sequentially and often have the same underlying structure (CHEN; LIU, 2018).

Formally, a task in CL can be defined as a tuple $t_i = (\mathbb{X}_i, \mathbb{Y}_i, P_i(x,y))$, where i is the task ID; $\mathbb{X}_i$ and $\mathbb{Y}_i$ are the input and label spaces for the function to be learned; and $P_i(x,y)$ represents the joint distribution. In this sense, we are interested in the predictive performance of a model on all tasks $\{t_i | \ 1 \leq i \leq T\}$ it was trained on, while the training is restricted to occur one task at a time, with no (or restricted) access to data from the previous and subsequent tasks.

As the amount of data available increases over the years and current machine learning (ML) systems still have poor ability to solve new tasks without being properly retrained, solutions that involve continual and multi-task learning will become more prevalent (REBUFFI *et al.*, 2017). Also, as deep learning techniques are the state-of-the-art for several tasks in areas such as computer vision and natural language processing (REN *et al.*, 2015; DEVLIN *et al.*, 2018), the adaptation of the ongoing strategies in these fields for the continual paradigm becomes a natural promising research direction.

Despite not being a new research topic (THRUN, 1995), there is still no consensus on all the characteristics that a CL model should consider essential (i.e., CL Desiderata) during its optimization process (ALJUNDI, 2019; MUNDT *et al.*, 2020). Most of the definitions favor a specific direction based on the researched topic the author is involved. For example, one may say that constant memory and forward transfer are fundamental for robotics. At the same time, for recommendation systems, one could argue that online learning and fast adaptation are more important features. Following this line of thought, for the continual object detection venue and especially the class-incremental setting, we argue that the following desiderata should be aimed:

- **Quasi-constant memory**: A CL model should work with bounded memory.
- **Backward Transfer**: A CL model should be able to improve the performance of previously learned tasks by learning a new one.
- **Forward Transfer**: A CL model should have the ability to improve the performance of future tasks using previously acquired knowledge.
- **Fast adaptation and recovery**: A CL model should be able to adapt quickly to new tasks, and, in case a class was gracefully forgotten (better described in the work of Ahn *et al.* (2019)), the model should recover the previous performance at the same speed.

Also, the ability to identify when a sample object is unknown at test time and decide whether to learn from it during incremental training is of interest for applications in autonomous robots (JOSEPH *et al.*, 2021). This scenario, which is related to other different ML paradigms (e.g., out-of-distribution detection, open-set, and open-world recognition), might be a pursued direction for having less human interference in the learning process (MUNDT *et al.*, 2020; MUNDT *et al.*, 2021).

2.2.1 Continual Learning Scenarios

When working with classical CL benchmarks (LOMONACO; MALTONI, 2017), there are three general situations in which data might be introduced:

- **New Instances (NI)**: New training samples of previously known classes.
- **New Classes (NC)**: Only training samples of new classes.
- **New Instances and Classes (NIC)**: New training samples from both old and new classes.

When working on classification tasks, the presence of the task ID dictates the space of possible classes and distributions that can be recognized during test time. Thus, it describes whether it is possible to create task-specific solutions or if a more general CL strategy is needed (DELANGE *et al.*, 2021). Following this trend, the CL literature has mostly adopted the convention from Ven and Tolias (2019) for three general task scenarios:

- **Task-Incremental Learning**: Assumes the model has information about the task ID during training and testing. The situation allows for task-specific solutions.
- **Domain-Incremental Learning**: Assumes the task ID is not given during test time, but the structure of the task is maintained. Class labels are usually kept, but the data distribution might change.
- **Class-Incremental Learning**: Assumes the task ID is not given during test time, and the model needs to infer it. In this way, the model needs to expand its range of predictions and incrementally add new classes.

Additionally, Task-Free or Task-Agnostic CL (ALJUNDI; KELCHTERMANS; TUYTE-LAARS, 2019; NORMANDIN *et al.*, 2021) represents an additional scenario for when the task labels are not given during either training or testing, which makes it the most challenging scheme. For that, the model does not have any information on task boundaries and still needs to deal with data distribution changes.

2.2.2 Continual Learning Evaluation

For evaluating CL models on incremental benchmarks, metrics should assess the desired characteristics we expect the system to have. To this extent, a CL model, in general, should be evaluated not only on its final performance but also on how transferable its knowledge is and how fast it learns and forgets tasks. The usual procedure adopted by the CL community to comply with this scheme was first introduced by Lopez-Paz and Ranzato (2017) with three metrics.

Average Accuracy (ACC) is the average final accuracy over all seen T tasks as described by Equation 2.1.

$$ACC = \frac{1}{T}\sum_{i=1}^{T} R_{T,i} \tag{2.1}$$

Backward Transfer (BWT), as shown by Equation 2.2, is the measure of the influence that learning a new task has on the tasks learned so far. A negative value for this metric indicates the forgetting of old classes.

$$BWT = \frac{1}{T-1}\sum_{i=1}^{T-1} R_{T,i} - R_{i,i} \tag{2.2}$$

Forward Transfer (FWT), as demonstrated in Equation 2.3, represents the impact that learning a new task will have on the consecutive tasks. A positive forward transfer is an indication that the model can perform “zero-shot” learning.

$$FWT = \frac{1}{T-1}\sum_{i=2}^{T} R_{i-1,i} - \bar{b}_i \tag{2.3}$$

For these metrics, $R_{i,j}$ stands for the final test accuracy on task t_j after observing the samples of task t_i, and $\bar{b}$ the test accuracy of each task when trained with random initialization. The metrics above assume the model has access to all tasks beforehand and can be evaluated on all T tasks right after it finishes the training in each individual task t_i.

For measuring how far an incremental model response is from an ideal setting and therefore assessing its overall stability-plasticity, Hayes *et al.* (2018) proposed Ω as the ratio between the model’s response and the one from the joint-training equivalent (i.e., a model trained offline with all task data) as shown by Equation 2.4. We will refer to this metric as the upper-bound ratio.

$$Upper\text{-}bound\ ratio\ (\Omega_{all}) = \sum_{t=1}^{T} \frac{R_{T,t}}{Rjoint,t} \tag{2.4}$$

Although there are interesting adaptations of these metrics that account for the performance of a CL model along each timestep in training time, in an application context, a good final performance at test time is usually what is considered. Additionally, some other metrics provide helpful information regarding the whole CL desiderata, such as computational efficiency and memory size (DÍAZ-RODRÍGUEZ *et al.*, 2018), but we will not explore them in the current context of this background review.

2.2.3 Continual Learning Strategies

Research to overcome catastrophic forgetting is as old as the own field of neural networks (ROBINS, 1995; RUMELHART, 1992), but previously had its focus on solving the problem for shallow networks. When dealing with deep architectures, the main methods have been commonly divided into three families of techniques based on: parameter isolation, regularization, and replay (DELANGE *et al.*, 2021). For a more in-depth description of each group of techniques, we recommend the reading of CL for classification specialized surveys and reviews (DELANGE *et al.*, 2021; BELOUADAH; POPESCU; KANELLOS, 2021; HAYES *et al.*, 2021).

2.2.3.1 Parameter isolation techniques

Parameter isolation strategies aim to mitigate forgetting by specifying parameters to deal with each individual task. This setup typically requires the freezing of some network parameters and then either dynamically expanding the network's capacity (YOON *et al.*, 2017) when new tasks arrive or learning specific sparse masks (MALLYA; DAVIS; LAZEBNIK, 2018).

One of the base works for this family was proposed by Rusu *et al.* (2016), where a deep neural network column of layers is trained to execute a single task. When a new task arrives, the previously trained weights are frozen, and a new column of layers with a lateral connection to the first column is added and then trained to execute the new task. Other works also expand on this strategy to deal with the issues caused by the increased final model size by applying network pruning and quantization (HUNG *et al.*, 2019). For this family of techniques, it is generally guaranteed that the network will perform equally well as if it was trained from scratch at the cost of having a more significant memory footprint. Additionally, models in this group often have the disadvantage of needing a task oracle to reveal the task ID at test time (DELANGE *et al.*, 2021).

2.2.3.2 Regularization-based techniques

Regularization-based methods introduce strategies to prevent the network parameters from deviating too much from the learned values that performed well for the old classes. The base work of Kirkpatrick *et al.* (2017) proposed the Elastic Weight Consolidation (EWC) strategy, which first finds important parameters for the learned tasks and then penalizes their changes when new tasks are presented.

Besides penalty-based regularization, Li and Hoiem (2017) suggested the Learning Without Forgetting (LwF) strategy in which a copy of the network trained on the base classes is created and knowledge distillation is applied to transfer the knowledge of the copy to the network trained on the new data. For this whole family of methods, there is generally no need for storing old data or changing the current architecture. This is based on the assumption that the task's knowledge is included in the weights and can be preserved by either penalizing their change directly or by constraining the updates for new data using the old activations and logits.

However, for this group of techniques, performance is often limited compared to other CL strategies (PELLEGRINI *et al.*, 2019; BEAULIEU *et al.*, 2020).

2.2.3.3 Replay techniques

Methods based on replay, often called rehearsal, store samples from previously seen data or use generative models to create pseudo-samples that follow the previous data distribution. The replay samples are then mixed with the ones of the new task to ensure that the data distribution of the new task does not deviate much from the previously learned data distributions. Following this line, Rebuffi *et al.* (2017) proposed the iCaRL strategy in which the samples that best represent the class means in the feature space are stored and used at test time with a nearest-mean classifier. In a different way, Lopez-Paz and Ranzato (2017) proposed the Gradient Episodic Memory (GEM) technique to constrain the model optimization by using replay samples to limit the gradients for the new task in a way that the approximated loss from the previous tasks will not increase.

When working with unstructured data (e.g., images and videos), the required memory buffer to store old samples might be considerably large, making its use impracticable for some real-world scenarios (PELLEGRINI *et al.*, 2019). Techniques based on pseudo-rehearsal, a.k.a. generative replay, were established to overcome this limitation. Shin *et al.* (2017) proposed to train a generative model on the old data distribution and use it to generate fake samples that help in mitigating the forgetting of old classes. Although having the downside of the model's performance being upper-bounded by the joint training in all tasks (DELANGE *et al.*, 2021), the replay family has been the most consistently used strategy in real-world applications of CL (SHAHEEN *et al.*, 2021; SHIEH *et al.*, 2020).

2.2.4 Other Continual Learning Paradigms

Some other learning paradigms have been adjusted to diminish the forgetting of CL systems by allowing the model to learn the desired adaptability and stability directly from the data (CACCIA; PINEAU, 2021; HOSPEDALES *et al.*, 2020).

2.2.4.1 Meta-Learning for Continual Learning

Meta-learning, a.k.a. "learning-to-learn", uses knowledge obtained from learning tasks to improve the learning of new ones. Because of the general terminology, there are several perspectives proposed in the literature that relate to the topic, such as transfer learning, AutoML, and multi-task learning (HOSPEDALES *et al.*, 2020). In the context of neural networks, meta-learning has been framed as an end-to-end pipeline with two levels where an outer algorithm adjusts the learning of an inner algorithm so that the outer model objective is improved in the end. In simpler words, it is the search for inductive biases in a neural network that leads to the fulfillment of a meta-level objective. This meta-objective can be applied for diverse goals such as generalization performance, fast adaptation, or even the avoidance of catastrophic forgetting (FLENNERHAG *et al.*, 2019).

The application of meta-learning to solve CL meta-objectives has been referred to as meta-continual learning (CACCIA *et al.*, 2020) and can take different forms. Rajasegaran *et al.* (2020) introduced the use of meta-learning for finding a set of generic weights that can generalize well for all seen tasks by quickly adapting to them at test time with minimum forgetting. Javed and White (2019) proposed a meta-objective for finding task-independent network representations that minimize the forgetting of old tasks and accelerate future learning of new ones. Beaulieu *et al.* (2020) presented the ANML strategy which uses a neuromodulatory network to modulate the learning of a base network by gating the neurons in a specific layer during the forward and backward passes.

2.2.4.2 Self-Supervision for Continual Learning

Self-supervision is the paradigm in which the data generates its own labels and learns to predict them back as a pretext task. Some examples of pretext tasks are colorizing grayscale images, predicting the rotation of objects, and matching different augmented views of the same image (HUANG *et al.*, 2021). The advantage of having the data generate its own supervision signal is to be able to explore large-scale unlabeled datasets and obtain robust representations that can be used for other downstream tasks such as image classification, object detection, and semantic segmentation (JING; TIAN, 2020). Recently, self-supervised pre-trained networks outperformed their supervised counterpart for downstream tasks of classification and detection in large benchmarks (CARON *et al.*, 2020; BAR *et al.*, 2021).

In the context of CL, the feature extraction backbone is generally frozen for not allowing gradual changes in the representations during online updates. This inevitably causes the need for networks that can produce more general features, which favors the use of self-supervision in their training. In fact, Gallardo, Hayes and Kanan (2021) showed empirically that self-supervised pre-trained models provide representations that generalize better for class-incremental learning scenarios, while Hu *et al.* (2022) also demonstrated effectiveness in applying self-supervision sequentially for learning representations from large-scale streaming data. Pham, Liu and Hoi (2021) proposed a learning structure based on the human brain complementary learning system, in which a model is optimized via self-supervision on stored samples to produce general representations that are then refined by supervised learning for quick knowledge acquisition on the labeled data. Beyond that, Caccia and Pineau (2021) expanded the generality of self-supervised representations to the meta-learning world by having models optimized to match different augmented views of the same image and at the same time generate representations that minimize the forgetting of old classes.

2.3 Object Detection with Deep Neural Networks

Object detection is a computer vision task that involves the localization and classification of items of interest in an image. The goal of an object detector is to predict the coordinates of each bounding box that surrounds the objects of interest and assign a category to it. Previous to 2012, most solutions related to the topic were based on heuristics and hand-crafted visual descriptors (VIOLA; JONES, 2001; LOWE, 2004) which limited its application in several domains. After the success that CNNs had in generating rich features for classification, they started to compose strategies for the more challenging task of object localization and recognition (GIRSHICK *et al.*, 2014; GIRSHICK, 2015). Since then, they have presented outstanding results in large competitions related to the detection task and became their baseline solution (WU; SAHOO; HOI, 2020).

Object detectors based on DNNs can usually be divided into two modalities: two-stage and one-stage detectors. Both have in common the presence of a backbone network for providing useful feature maps to be used in localization and identification of object categories (HUANG *et al.*, 2021). These features can be resumed in a single 3D tensor extracted directly from the output of a single layer in a pre-trained architecture (e.g., C4 layer in ResNet-50) or a multi-dimensional tensor resulting from the gathering of the output of several layers from a top-down architecture with lateral pathways as in the work of Lin *et al.* (2017a). The backbones used for detection tasks are generally deep CNNs pre-trained on large image datasets (e.g., ImageNet) intended for classification (DENG *et al.*, 2009).

2.3.1 Two-Stage Detectors

This class of detectors uses a separate structure to generate a set of "guesses" of where the objects are present in the image. These assumptions on the image, also called region proposals or just proposals, will be then classified into the known categories and have their bounding box refined to correctly identify the object's limits. R-CNNs (GIRSHICK *et al.*, 2014) were one of the first two-stage strategies for object detection and used Selective Search (UIJLINGS *et al.*, 2013) for selecting its region proposals. The problem with this setup was that every proposal was processed separately by the CNN for feature extraction, which caused the inference process to be too slow. In the following work of the same authors, they propose the Fast-RCNN (GIRSHICK, 2015) in which a CNN first processes the image to extract the features maps. Then, the external proposals are used to select the regions within the feature maps through a Region of Interest (RoI) pooling layer, to be processed by the classification and regression heads as illustrated in Figure 6.

In the work of Ren *et al.* (2015), the authors ceased the use of heuristics for selecting region proposals by using a separate network called Region Proposal Network (RPN) able to be optimized specifically for identifying more probable regions of objects within an image. Their

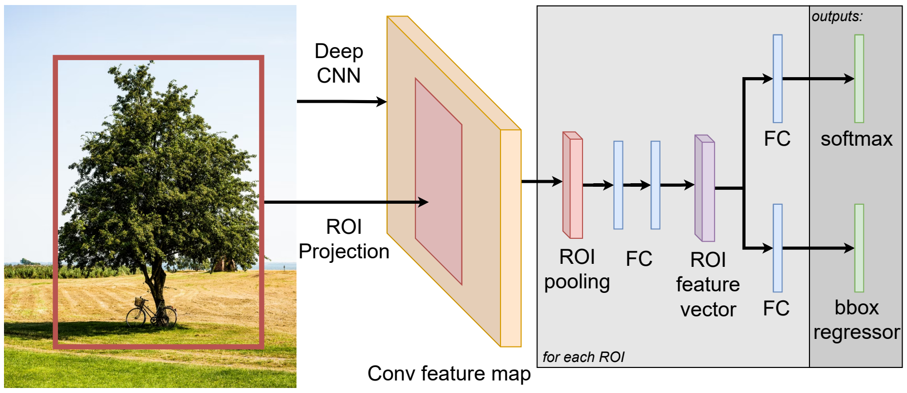


Figure 6 – Fast-RCNN architecture (GIRSHICK, 2015).

solution used the same structure as Fast-RCNN. Still, it was way faster than its counterpart, which resulted in it being named Faster-RCNN. Lin *et al.* (2017a) improved the network backbone performance in generating robust features to identify smaller objects. Their strategy, called Feature Pyramid Networks (FPN), exploited the "inherent multi-scale pyramidal hierarchy" that deep CNNs carry through exploring a top-down architecture with lateral connections that helps in the propagation of information from the higher layers to the lower ones. An illustration of a Faster-RCNN with FPN can be seen in Figure 7

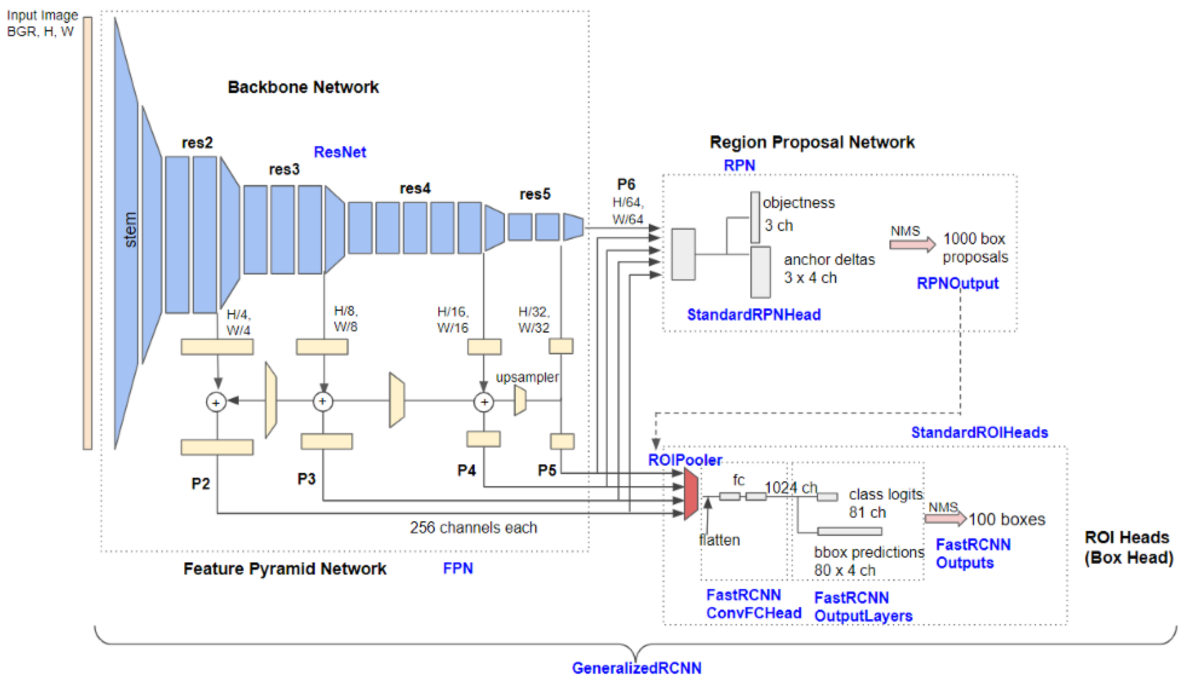
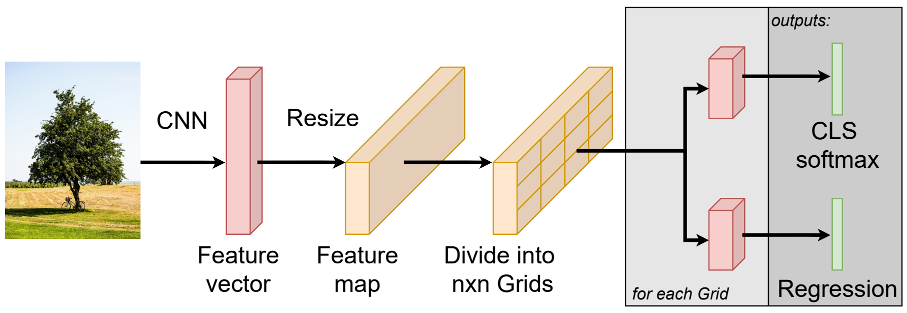


Figure 7 – A Faster-RCNN with FPN (HONDA, 2022).

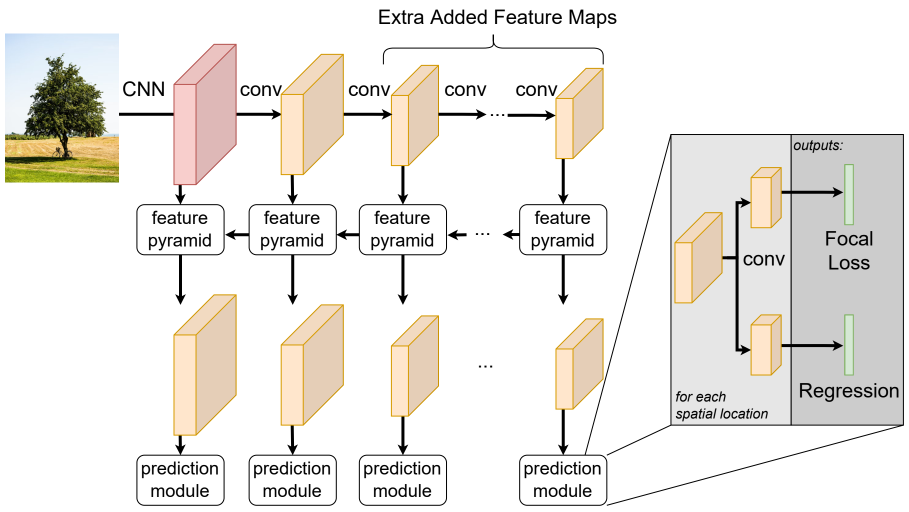
2.3.2 *One-Stage Detectors*

One-stage models, also known as single-stage detectors, are often faster than their two-stage counterparts at the cost of having lower predictive performance (HUANG *et al.*, 2021). There is no region proposal heuristic or network for this class of models since it usually considers that every position on the image might have an object, leaving the model to classify each position as either background or the target category. The You Only Look Once (YOLO) detector (REDMON *et al.*, 2016) was one of the first successful models to show a good balance between accuracy and speed by dividing the whole image into a set of grid cells and predicting the presence of one or more objects in each of them.

Improving on the inferior ability of the first YOLO architecture for detecting smaller objects, Liu *et al.* (2016) proposed the Single Shot Multibox Detector (SSD), which made use of a more elaborated CNN architecture and a set of pre-defined anchors in multiple scales and aspect-ratios. These additional features helped the model reach a decent performance while still operating in real time. Building on top of that, Lin *et al.* (2017b) presented RetinaNet, which focused on dealing with the large number of negative samples that are generated by the pre-defined anchors using their Focal Loss. This loss weights down the importance of easy negative samples while increasing the focus of the network weight updates on the hard ones. The network also uses FPN in its architecture and has reached results that compare to Faster-RCNN. An illustration of the general pipeline used in the YOLO and RetinaNet detectors is shown in Figure 8.



(a) YOLO



(b) RetinaNet

Figure 8 – The general pipeline throughout the YOLO and RetinaNet architectures.

Later on, several versions of the YOLO architecture, which are commonly referred to as the “YOLO family”, have been proposed and optimized for decreasing the gap against two-stage models regarding *mAP* performance (GE *et al.*, 2021) while keeping the real-time characteristic. Moreover, recently a few more elaborated strategies, such as CenterNet (DUAN *et al.*, 2019) and FCOS (TIAN *et al.*, 2020), that do not make use of either pre-defined anchor boxes or proposals, have raised the bar for the performance in popular detection benchmarks.

2.3.3 Benchmarks

Training large DNNs requires the availability of large datasets since they tend to be more accurate as more data gets processed (LECUN; BENGIO; HINTON, 2015). Considering that annotations for detection are harder to be obtained than just labels for the whole image, the most popular benchmarks on the topic have become the ones from competitions organized by resourceful universities or big tech companies. The two most explored are the Pascal VOC (EVERINGHAM *et al.*, 2010) and MS COCO (LIN *et al.*, 2014). Although there are different versions of the datasets based on the year of the challenges, researchers have adopted the VOC 2007 and COCO 2014 as references. Table 1 displays some statistics related to these benchmarks.

Table 1 – Statistics for the main object detection benchmarks (ZOU *et al.*, 2019).

Dataset	VOC 2007	COCO 2014
Number of classes	20	80
Number of training images (train+val)	5,011	123,287
Number of training instances	12,608	896,782
Number of testing images	4,952	81,434
Mean of bounding boxes per each training image	2.51	7.27

Recently, the LVIS dataset (GUPTA; DOLLAR; GIRSHICK, 2019) was released with the promise of being a more complex (and natural) benchmark due to its vast number of categories but a low amount of samples in some of them. The dataset has over 164,000 images with more than 1000 categories and a total of 2.2 million high-quality annotations, which makes it a tough challenge for generalization on the “long-tailed” categories.

2.3.4 Evaluation

The evaluation of object detection models is conducted by assessing how much each predicted bounding box misses or hits a ground truth based on a threshold. The equation that governs this metric is the intersection over union (IOU), also known as the Jaccard Index (Equation 2.5) in which B_{pred} is the coordinate of the predicted bounding box and B_{gt} is the ground truth equivalent (PADILLA; NETTO; SILVA, 2020). An illustration of these terms is shown in Figure 9.

$$\text{Jaccard Index} = IOU = \frac{area(B_{pred} \cap B_{gt})}{area(B_{pred} \cup B_{gt})} \tag{2.5}$$

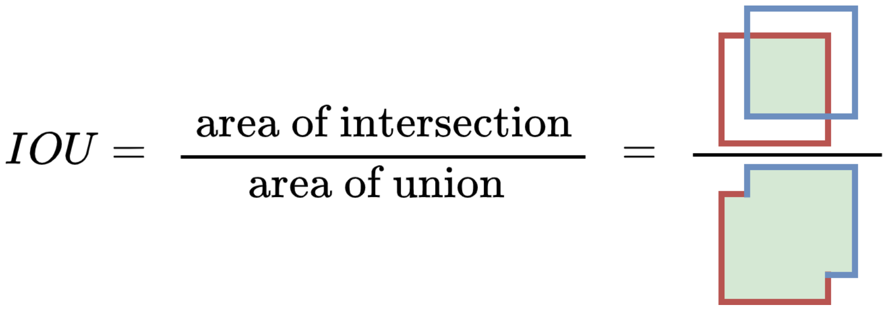


Figure 9 – Illustration of the Intersection Over Union equation. Image adapted from Padilla, Netto and Silva (2020).

The threshold value indicates how much overlap is needed to consider that a prediction was, in fact, a true positive. Then, the comparison of detection models can be made by calculating the average precision (AP) (i.e., the ratio of true positives over the sum of true positives and false positives) and average recall (AR) (i.e., the ratio of true positives over the sum of true positives and false negatives) for a given threshold. Equations 2.6 and 2.7 describe both metrics.

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{all\ detections} \tag{2.6}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{all\ ground\ truths} \tag{2.7}$$

A common value used for the threshold is 0.5 (e.g., AP^{50}). The standard evaluation procedure is to consider the mean average precision (*mAP*) at a given threshold for all classes that a detector is able to recognize. Moreover, for better dealing with false negatives, the *mAP* term is commonly assigned as the area under the curve (AUC) of the precision against the recall curve using the specified threshold (PADILLA; NETTO; SILVA, 2020). In addition to that determination, for some situations, benchmarks may also use the mean over the average precision of each class for several thresholds (e.g., $mAP@[.5:.95]$) (LIN *et al.*, 2014) to indicate a more stable performance.

CHAPTER

3

CONTINUAL OBJECT DETECTION: A LITERATURE REVIEW

In this chapter, we present the results of a thorough systematic review of the class-incremental object detection methods, benchmarks, and evaluation procedures. As a reference, the following review was accepted for publication in the Neural Networks journal in 2022 and took into account all important works that had been done in the field up to March 2022.

3.1 Introduction

The general goal of the continual learning paradigm for object detection is to learn a sequence of tasks $[t_1, t_2, t_3, \ldots]$ and have a model able to successfully localize and identify all the involved classes from the tasks at test time as illustrated by Figure 10.

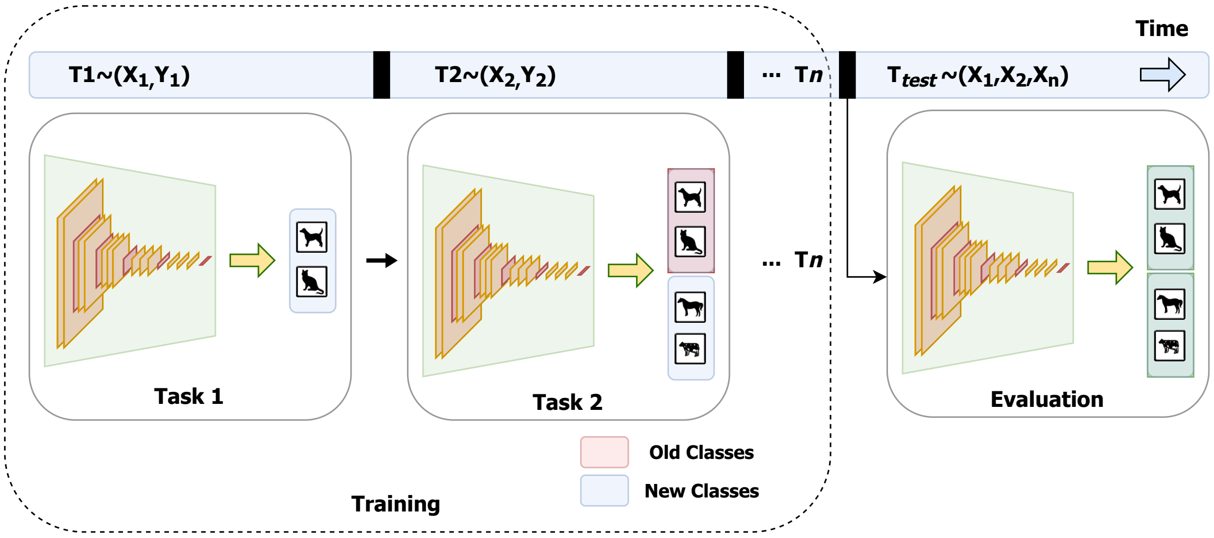


Figure 10 – A generic class-incremental scenario for object detection.

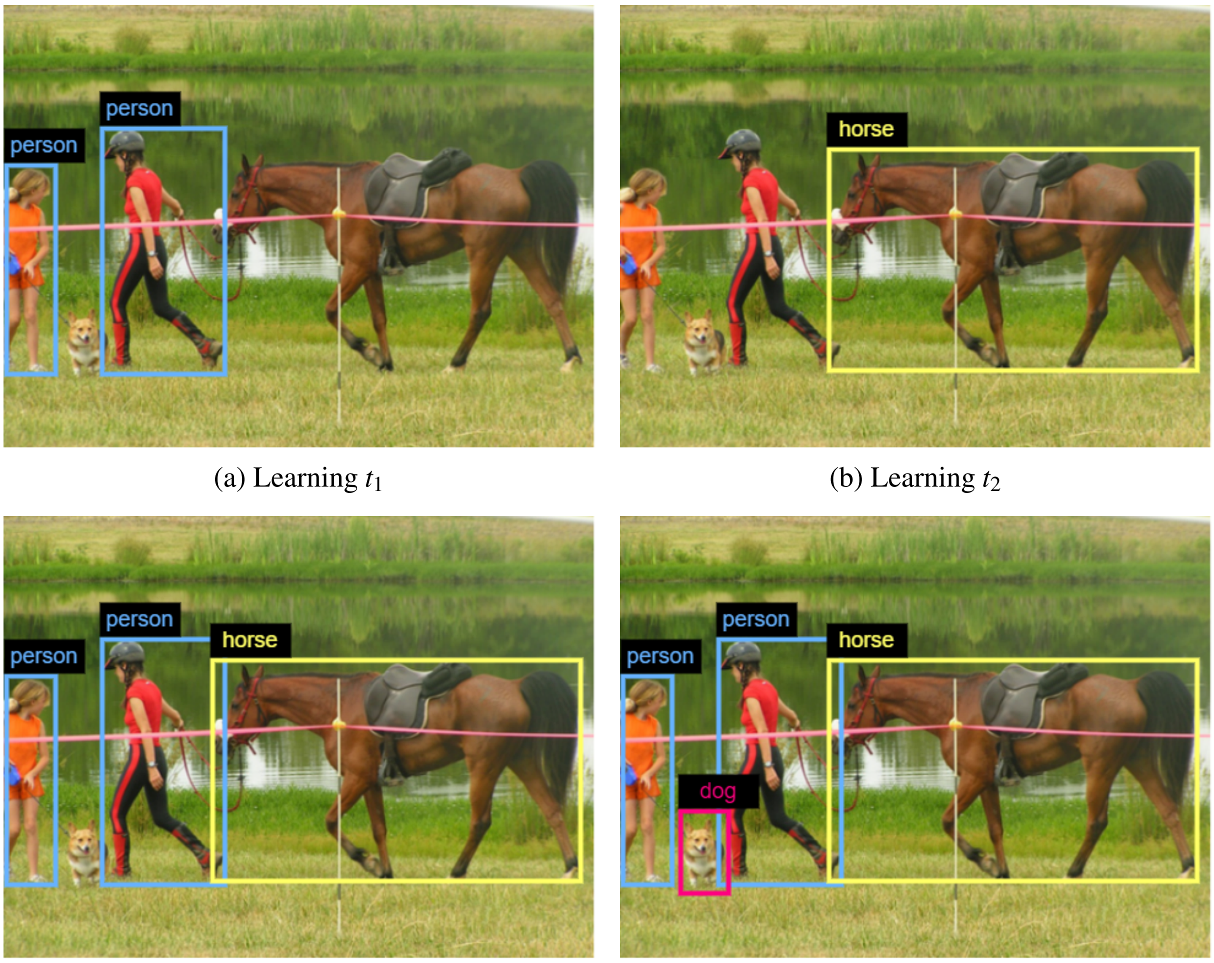
The area of applications of continual learning methods to object detection is still young and in active development (SHMELKOV; SCHMID; ALAHARI, 2017; PENG; ZHAO; LOVELL, 2020; YANG; ZHOU; WANG, 2021). Strategies developed so far are mostly split into two large pools: Class-Incremental Object Detection (CIOD) and Domain-Incremental Object Detection (DIOD). The former looks at problems where the model has learned the representation of base classes and then needs to extend its prediction power over new unknown classes sequentially. The latter is formed by solutions to problems where the classes are fixed, but their distribution can change over time. In this situation, the model needs to be able to identify the classes in both contexts correctly (KUNDU *et al.*, 2020).

For DIOD, a recent competition showed through their winning solutions that general strategies that account mainly for classification biases might suffice (e.g., simple random replay, using larger networks) even in challenging scenarios (LI *et al.*, 2022; ACHARYA; KANAN, 2021; ZHAI; LIU, 2021). For that, we advise the reader to analyze the general findings and discussions present in related surveys and review papers that reference this CL setting (HADSELL *et al.*, 2020; PARISI *et al.*, 2019; DELANGE *et al.*, 2021). Contrastively, we argue that the CIOD paradigm needs a more specific treatment due to its inherent challenges and complexity

The task of incrementally adding classes to a trained detector is considered of substantial importance for several applications that deal with memory and computational constraints (SHAHEEN *et al.*, 2021). The main issue that makes detection a more difficult task than only classification for class-incremental scenarios is that the same image can have several instances of different objects that are unknown apriori. Since these objects are not identified, the network learns to treat their visual cues as background instances. Later, when images of the unknown instances present before are shown as a new class, the model tends to either not converge to a decent solution or only prioritize the learning of the new category. In other words, this label conflict favors the interference on the weights specific to each task within the network.

Figure 11 exemplifies the process of incrementally learning some classes in sequence. Figures 11a and 11b show an example of two classes being learned separately, whilst Figure 11c shows the new class from task t_2 being learned after t_1. At last, Figure 11d shows a third class from task t_3 being added to the model. To exemplify why CIOD is considered a harder task than classification, the class “person” for the first learning task represented in Figure 11a is considered as background on the second task annotations as shown by Figure 11b. This naturally results in a label conflict that might induce catastrophic forgetting and harm the final detection performance.

Although still in its first steps, the CIOD field has a more established corpus of strategies, and some of them can also be applied within the domain-incremental option (KUNDU *et al.*, 2020; LI *et al.*, 2022). The first proposed strategy for CIOD dates back to 2017 in the seminal paper written by Shmelkov, Schmid and Alahari (2017). Since then, several methods have been presented with the goal of tackling forgetting while making DNNs localize and recognize classes



(a) Learning t_1

(b) Learning t_2

(c) Incrementally learning t_2 after t_1

(d) Incrementally learning a t_3 after $[t_1, t_2]$.

Figure 11 – Examples of learning separately some task t_1 and t_2; and incrementally learning $[t_1, t_2, t_3]$.

incrementally. For a more concise way of analyzing all the recent contributions to this field, we performed a systematic review of all the papers that included evaluations within the scope of continual object detection for class-incremental scenarios.[1]

3.2 Considerations about the Literature Review

For gathering the most influential work related to the CIOD field, we took advantage of the fact that the initial paper of Shmelkov, Schmid and Alahari (2017) presented a solid baseline for the problem, which indirectly guided the field to always make comparisons to it. In this way, we chose to perform a snowballing literature review followed by the guidelines described on Wohlin (2014). In this review technique, a paper (or a set of papers) has its citations and references explored in a forward and backward iterative process in order to find all works

[1] Even though the most used term in the literature for models that are able to detect new classes incrementally is "incremental object detection", we adopt as a reference in this Ph.D. thesis a more specific treatment (i.e., CIOD) to not confuse with strategies that only deal with domain-incremental scenarios.

that deal with the topic of interest. A general description of the review pipeline is described in Figure 12.

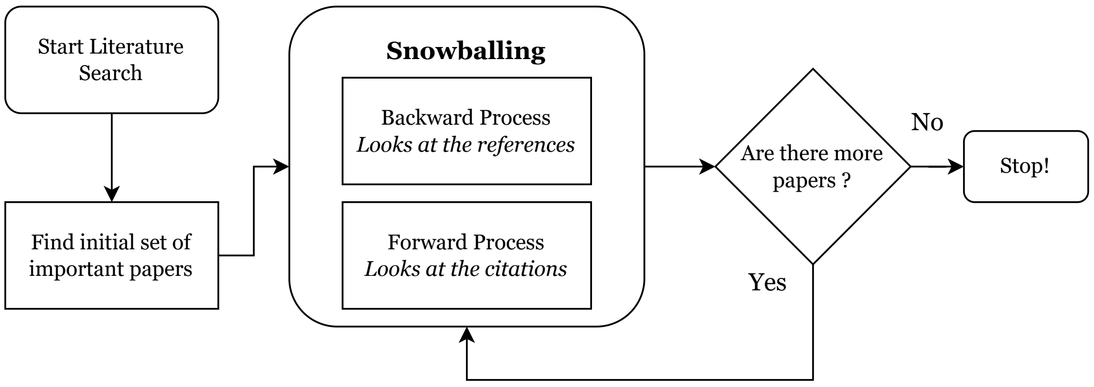


Figure 12 – The adopted snowballing review process.

Since the research field is reasonably new, some relevant work will certainly be placed first on arXiv as pre-prints. Because of that, we decided to use the Google Scholar database for checking the citations and references since they aggregate all the results from pre-print sources (e.g., arXiv and bioRxiv) to several popular scientific databases such as IEEE Xplore, ACM Digital Library, Scopus, and Science Direct.

3.2.1 Research Questions

With this review, we aimed to answer the following research questions regarding CIOD:

RQ1: What are the main benchmarks?

RQ2: What are the main metrics?

RQ3: What are the main proposed strategies and their differences?

RQ4: What is the current state-of-the-art with respect to performance?

3.2.2 Inclusion and Exclusion criteria

For starting the forward and backward process inherent to the snowballing technique, we considered the following inclusion criteria:

✓ Papers that cited Shmelkov, Schmid and Alahari (2017) or appeared in its reference list.

Then, we iteratively checked all the papers that made citations (up to March 2022) or appeared in the reference list of the first pool of gathered papers and proceeded in a loop until no more studies could be considered. At the same time, for selecting the works that mattered to this proposal from this large set, we established the following exclusion criteria:

× Paper was not written in English.

× Paper did not propose a technique, benchmark or metric related to the CIOD paradigm.

× Paper did not go under the peer-review process or, if published as pre-print online, did not have citations.

As stated above, we adopted the requirement for citations as a quality measure only for the works published as pre-prints online. This strategy was adopted considering that the CIOD field is recent (i.e., many papers are placed as pre-prints before being published), and we value public acceptance as a way to evaluate the paper‘s integrity. After analyzing all related work, 26 research papers followed the criteria and provided answers to the aspects indicated by the aforementioned research questions.

3.2.3 Literature Review Results

In this section, we proceed with the discussion of the review results and the formulation of answers to each research question.

3.2.3.1 RQ1: What are the main benchmarks ?

To account for the realism assumption needed for CL, researchers have focused on adapting current large-scale object detection benchmarks to the class incremental setting. Differently from CL for classification, where images are labeled with only one class and are presented to the model once, for CIOD, one image can have several objects with some (or several) of them unknown at the annotation time.

As an alternative to simulate this scenario, researchers modify the model‘s dataloader, allowing it to see only images with objects from the classes of interest, making sure to omit the annotations related to the classes that are not part of the current task. Thus, the likelihood an image will be presented to the model more than once depends on the number of annotated classes in the image and which classes are included in the task. Intuitively, for this setting, the models face an extreme situation where the training data for the subsequent tasks might not share the same label space as the previous tasks. As a result, the model sees the same image with different annotations, making it difficult to find feature representations that generalize well across previous and future tasks.

Most of the CIOD strategies that are presented in Sections 3.2.3.3 were compared using the adaptation of the traditional VOC and COCO benchmarks, with the introduction of classes sequentially in single units or pre-defined groups of multiple classes. One caveat of this setting is that the choice of classes that are part of a task impacts the model‘s predictive performance. For example, as discussed by Peng, Zhao and Lovell (2020), when working on the VOC benchmark (20 classes) with a training setting of $19+1$ (i.e., learning 19 classes at once and one incrementally), learning the class "person" incrementally is a harder task than learning the object "TV". This occurs because the person object appears more times (i.e., has more images associated with it) and constantly occurs with other category objects in the same image, which, for this learning scenario, results in 40.87% of missing annotations.

For standardization, the incremental versions of the VOC dataset are generated by sorting the classes alphabetically and generally splitting the dataset into four different scenarios, as illustrated by Figure 13.

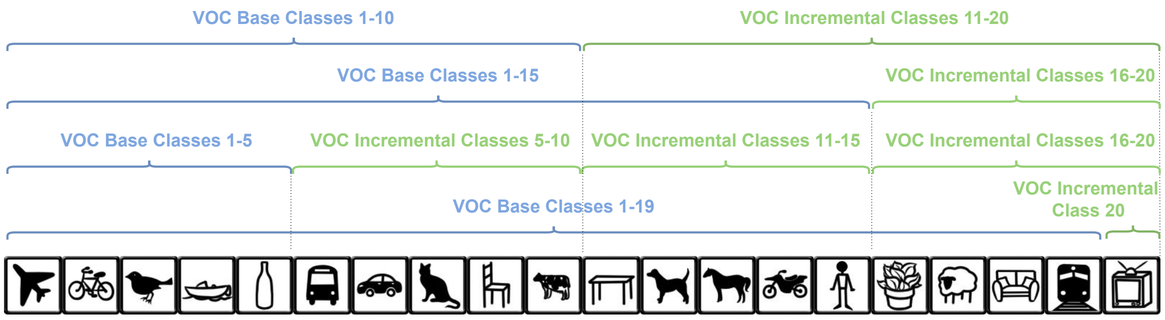

VOC Base Classes 1-10
VOC Incremental Classes 11-20
VOC Base Classes 1-15
VOC Incremental Classes 16-20
VOC Base Classes 1-5
VOC Incremental Classes 5-10
VOC Incremental Classes 11-15
VOC Incremental Classes 16-20
VOC Base Classes 1-19
VOC Incremental Class 20

Figure 13 – Description of some of the adopted incremental scenarios for the Pascal VOC 2007 dataset.

For the incremental scheme of the COCO dataset, classes are ordered following the ID of the original labels and usually split in half to create a unique scenario with 40 classes for training the base model and 40 to be added sequentially at once as shown in Figure 14.

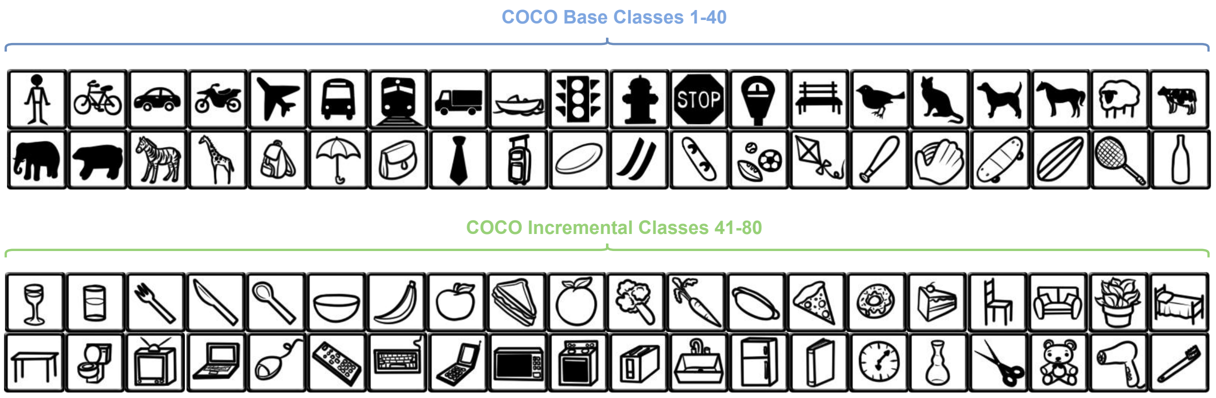


Figure 14 – Description of an incremental scenario with MS COCO 2014 dataset.

For fair comparisons using the two popular large-scale benchmarks, researchers usually train the network using 40k and 400k iterations, respectively, for the VOC and COCO when learning the first task with the most classes. In the following incremental steps, the model is trained using 5k-10k iterations when only one class is added and the same number of iterations as the first step if learning multiple classes at once.

Besides the aforementioned benchmarks, some authors used the same incremental-style adaptation with popular and private large-scale datasets specific to other tasks, such as remote sensing (DOTA and DIOR) (CHEN *et al.*, 2020) and autonomous driving (ITRI-DriveNet-60 and KITTI) (SHIEH *et al.*, 2020; RAMAKRISHNAN *et al.*, 2020).

Although not well explored, there were a few benchmarks specifically designed for evaluating CIOD solutions. Hao, Fu and Jiang (2019) introduced a large-scale dataset of vending machine products, named Take Goods from Shelves (TGFS), with 38k images and 24 possible categories. This benchmark also has three coarse classes that cover the categories meant to instigate class-incremental detection solutions to retail problems. However, until the date of this review was written, it was not publicly available. Wang *et al.* (2021b) proposed an egocentric video dataset that focused on capturing objects and scenes present in the daily life of a university student. The benchmark, named Objects Around Krishna (OAK), was created for online continual object detection tasks. For the online continual setting, a continuous temporal stream (i.e., video) is given to the model, which can only learn from it through one training experience. Afterwards, some frames are extracted and used for continual evaluation of the model's performance in the subsequent learning experiences. As new annotated objects arrive, the model needs to learn them incrementally throughout the streaming experience. Considering the reality aspect of the task, this benchmark brings a more plausible and difficult scenario for a real-world mobile robot.

3.2.3.2 RQ2: What are the main metrics ?

The evaluation for CIOD has followed the same structure of traditional object detection with the use of $mAP@.5$ for nearly all benchmarks and $mAP@0.5-0.95$ for COCO like datasets. However, some researchers noticed that directly comparing the mAP performance of techniques on the same benchmark would not assess their real efficiency since changes in the training regime and even framework could cause the same method to present different results. To comply with that, the difference and the ratio against the upper-bound (i.e., joint-training with all classes at once) have been commonly used for comparisons (LIU *et al.*, 2020; ACHARYA; HAYES; KANAN, 2020) since they represent how the performance would be in case data could be fully accumulated and create a common ground between techniques (i.e., how far we are from the ideal response). Yet, the gap against the joint-training is only meaningful when both methods are implemented within the same training regime and framework since only in this situation it is possible to ascertain which single components really contributed to narrowing the gap. Most researchers do not consider this setting and pick up results from different papers to compare against their joint-training outcomes, which does not give more information than checking their single mAP results.

Beyond that, Chen, Yu and Chen (2019) proposed the use of a F_{map} metric, inspired by the $F_1-score$, in which they calculate the harmonic mean between the mAP values of old and new classes as described by Equation 3.1.

$$F_{map} = \frac{2\ mAP_{old}\ mAP_{new}}{mAP_{old} + mAP_{new}} \tag{3.1}$$

Yang, Zhou and Wang (2021) introduced a metric called Stability-Plasticity-mAP *SPmAP* that considers how much the incremental learning process affects the average stability and plasticity of a detector. Their metric takes into consideration the mean differences of the incremental model against the upper-bound for the old and new classes as shown by Equation 3.2.

$$SPmAP = \frac{\frac{Stability+Plasticity}{2} + mAP_{dif}}{2} \tag{3.2}$$

$$Stability = \frac{1}{N_{old\_classes}} \sum_{i=1}^{N_{old\_classes}} (mAP_{joint,i} - mAP_{inc,i})$$

$$Plasticity = \frac{1}{N_{new\_classes}} \sum_{i=N_{old\_classes}+1}^{N_{all\_classes}} (mAP_{joint,i} - mAP_{inc,i})$$

$$mAP_{dif} = \frac{1}{N_{all\_classes}} \sum_{i=1}^{N_{all\_classes}} (mAP_{joint,i} - mAP_{inc,i})$$

We also believe that CIOD models can only be compared when the join-training results of their architecture are available. However, only looking at the discrepancy between the incremental and joint-training models does not lead to the understanding of which specific aspects of the strategy are failing. The aforementioned metrics are helpful, but they lack the specificity for identifying where the incremental model should pay attention. To circumvent that, we propose two separate metrics that compare and scale the final incremental *mAP* values for each class against the joint-training separately for the old and new categories. These metrics are defined as the rate of stability (RSD) and plasticity (RPD) deficits as described in Equations 3.3 and 3.4.

$$\text{RSD} = \frac{1}{N_{old\_classes}} \sum_{i=1}^{N_{old\_classes}} \frac{mAP_{joint,i} - mAP_{inc,i}}{mAP_{joint,i}} * 100 \tag{3.3}$$

$$\text{RPD} = \frac{1}{N_{new\_classes}} \sum_{i=N_{old\_classes}+1}^{N_{new\_classes}} \frac{mAP_{joint,i} - mAP_{inc,i}}{mAP_{joint,i}} * 100 \tag{3.4}$$

Our metrics allow the direct interpretation of how much an incremental model compares to the upper-bound in remembering old classes and learning the new ones (e.g., the model has a 10% worse performance for recognizing previous classes, but only a 2% deficit for learning new categories when compared to the upper-bound). Therefore, for this context, a CIOD strategy should aim not only to reach a decent final *mAP* value and high upper-bound ratio but also to keep low and balanced stability and plasticity deficits. Additionally, the ratios can assume negative values, indicating that the incremental model has performed better than joint-training for some classes and reinforcing the relationship with standard CL metrics such as BWT. We applied the rate of plasticity and stability deficits along with the upper-bound difference during the performance analysis of Section 3.2.3.4.

3.2.3.3 RQ3: What are the main proposed strategies and their differences ?

For dissecting the contributions of each paper, we evaluated the selected works on their choice of strategy to mitigate forgetting, used architecture and backbone, benchmarks, and evaluation methods. The results are presented in Table 2 with some colored cells to aid in the analysis.

Additionally, to better examine the main contributions of each paper, we organized their takeaways primarily based on the type of chosen strategy to prevent forgetting. Considering that several papers applied more than one, we clustered them based on the strategy pointed to be their main highlight and attempted to present the takeaways for each selected topic in chronological order.

Knowledge Distillation

Knowledge Distillation, as mentioned previously, can be used as a regularization method by enforcing the incremental model in task t_{i+1} to account for the model's previous states and outputs obtained when training for task t_i (LI; HOIEM, 2017).

Shmelkov, Schmid and Alahari (2017) introduced the first work to deal with the CIOD problem through the use of "vanilla" knowledge distillation. The authors adapted the Fast-RCNN architecture to learn incrementally by using a copy of the network trained on the base classes as the teacher and another as the student. The teacher has its weights frozen and the student has to not only detect the newly introduced categories but also repeat the distribution of responses of the frozen teacher. This behavior is achieved by using an additional regularization loss based on the bounding box predictions and logits produced by both networks, inspired by the work of Li and Hoiem (2017). They chose to go with a strategy that used external proposals instead of learned ones because of their supposed class-agnostic robustness (GIRSHICK, 2015). Since they were the first to propose a strategy for this problem, most consecutive papers built solutions on top of their regularization approach and compared them to it.

Hao *et al.* (2019) adapted the Faster-RCNN architecture to the CIOD context with the expansion of the RPN to consider the new class as foreground. They evaluated the classification results using a fully connected network and a nearest prototype classifier. Additionally, they artificially avoided the possibility of background label conflict between old and new data by excluding images that contained objects from multiple class groups, which is unreal for a real-world setting. In a similar strategy, Chen *et al.* (2020) expanded the RPN for dealing with new classes and used knowledge distillation on the outputs of a teacher network to allow the model to detect remote sensing objects incrementally with minimum forgetting using the specific domain datasets proposed by Li *et al.* (2020) and Xia *et al.* (2018). Zhou *et al.* (2020) applied

Table 2 – Class-Incremental Object Detection main papers

References	Strategy	Benchmark	Backbone	Object Detector	Evaluation
Shmelkov, Schmid and Alahari (2017) (ILOD)	Knowledge Distillation	VOC 2007 COCO 2014	ResNet-50	Fast-RCNN	Multiple Classes Sequential Classes
Li *et al.* (2018) (MMN)	Parameter Isolation	VOC 2007	VGG-16	SSD-300	Multiple Classes Sequential Classes
Guan *et al.* (2018)	Pseudo-Labels	VOC 2007 TSD-MAX	Darknet-19	Yolo-V2	Multiple Classes
Hao *et al.* (2019) (CIFRCN)	Knowledge Distillation	VOC 2007 COCO 2014	ResNet-101	Faster-RCNN + Nearest Neighbor	Multiple Classes Sequential Classes
Chen, Yu and Chen (2019)	Knowledge Distillation	VOC 2007	ResNet	Faster-RCNN	Multiple Classes Sequential Classes
Li *et al.* (2019) (RILOD)	Knowledge Distillation External Data	VOC 2007 iKitchen	ResNet-50	RetinaNet	Multiple Classes Sequential Classes
Hao, Fu and Jiang (2019) (FCIOD)	Knowledge Distillation Replay	TGFS	ResNet-101	Faster-RCNN	Multiple Classes
Liu *et al.* (2020) (IncDet)	Pseudo-Labels EWC	VOC 2007 COCO 2014	ResNet-50	Fast-RCNN Faster-RCNN	Multiple Classes Sequential Classes
Acharya, Hayes and Kanan (2020) (RODEO)	Replay	VOC 2007 COCO 2014	ResNet-50	Fast-RCNN	Sequential Classes
Peng, Zhao and Lovell (2020) (Faster ILOD)	Knowledge Distillation	VOC 2007 COCO 2014	ResNet-50	Faster-RCNN	Multiple Classes Sequential Classes
Zhou *et al.* (2020)	Knowledge Distillation	VOC 2007 COCO 2014	ResNet-50	Faster-RCNN	Multiple Classes Sequential Classes
Zhang *et al.* (2020) (DMC)	Knowledge Distillation External Data	VOC 2007	ResNet-50 / ResNet-34	RetinaNet	Multiple Classes
Liu *et al.* (2020) (AFD)	Knowledge Distillation Replay	KITTI / Kitchen VOC 2007 COCO 2014 Comic / Watercolor	SE-ResNet-50	Faster-RCNN	Multiple Classes
Yang *et al.* (2020)	Pseudo-Labels Knowledge Distillation	VOC 2007 COCO 2014	ResNet-50	Faster-RCNN	Multiple Classes Sequential Classes
Shieh *et al.* (2020)	Replay	VOC 2007 ITRI-DriveNet-60	Darknet-53	Yolo-V3	Multiple Classes
Ramakrishnan *et al.* (2020) (RKT)	Knowledge Distillation	VOC 2007 VOC 2012 KITTI	ResNet	Fast-RCNN	Multiple Classes Sequential Classes
Chen *et al.* (2020)	Knowledge Distillation	DOTA / DIOR	Custom with FPN	Custom with two stages	Multiple Classes
Peng *et al.* (2021) (SID)	Knowledge Distillation	VOC 2007 COCO 2014	ResNet-50	CenterNet FCOS	Multiple Classes Sequential Classes
Joseph *et al.* (2021) (ORE)	Pseudo-Labels Replay	VOC 2007	ResNet-50	Faster-RCNN + Nearest Neighbor	Multiple Classes
Yang, Zhou and Wang (2021)	Knowledge Distillation	VOC 2007 COCO 2014	ResNet-50	Faster-RCNN	Multiple Classes Sequential Classes
Yang *et al.* (2021)	Replay	VOC 2007	ResNet-50	Faster-RCNN	Multiple Classes
Kj *et al.* (2021) (Meta-ILOD)	Knowledge Distillation Replay Meta-Learning	VOC 2007 COCO 2014	ResNet-50	Faster-RCNN	Multiple Classes Sequential Classes
Haq *et al.* (2021)	Knowledge Distillation	VOC 2007	Darknet-53	Yolo-V3	Sequential Classes
Zhang *et al.* (2021)	Parameter Isolation	VOC 2007	Darknet-53 + ResNet	Yolo-V3	Sequential Classes
Dong *et al.* (2021)	Knowledge Distillation External Data	VOC 2007 COCO 2014	ResNet-50	Faster-RCNN	Multiple Classes Sequential Classes
Wang *et al.* (2021b)	-	OAK	ResNet-50	Faster-RCNN	Sequential Classes

distillation on the detection heads and RPN outputs along with a supplementary sampling strategy to select proposals that tend to be from the foreground classes. Ramakrishnan *et al.* (2020), on the other hand, hypothesized that the relationship between region proposals and the ground truth annotations encoded the detector‘s knowledge. In this way, the authors introduced a strategy to select proposals based on this relation and applied distillation on the filtered samples. Haq *et al.* (2021) evaluated distilling knowledge only on the logits for the YOLO-V3 architecture in a setting with only two classes and showed better results than other CL strategies.

Beyond the basic distillation of the detector outputs, several methods proposed additionally to distill intermediate features of the base model. Chen, Yu and Chen (2019) presented the first work that made use of this type of distillation through what they named a "hint loss". Their final loss also considered the prediction confidence of the initial model for regularization, but the final performance seemed to suffer from plasticity issues. Peng, Zhao and Lovell (2020) made use of the Faster-RCNN and introduced an additional adaptive distillation step on the features and RPN outputs. They further investigated the negative impact that having old class objects within the new class images has on the performance of the RPN and concluded that it was not that significant, which explains why Faster-RCNN networks generalize better than solutions with external proposals. Peng *et al.* (2021) presented the use of distillation not only on intermediate features, but also on the relations (distances) between features of different samples for anchor-free object detectors. Yang, Zhou and Wang (2021) proposed the preservation of channel-wise, point-wise, and instance-wise correlations between some feature maps of the teacher and student networks in order to maintain the performance on the old classes while optimizing for the new ones.

Replay

Replay strategies usually prevent forgetting by storing samples (or trained generative models) from each previous task to balance the training for the subsequent tasks. In this sense, Hao, Fu and Jiang (2019) employed the use of a small buffer of samples along with logits distillation to perform better than its competitors in the incremental learning of common objects from vending machines. Shieh *et al.* (2020) proposed the use of experience replay with different buffer sizes and the YOLO-V3 architecture for the problem of adding multiple classes at once to an object detector. They evaluated their approach in a common benchmark and on a private autonomous driving dataset. Acharya, Hayes and Kanan (2020) suggested using product quantization to compress feature maps without losing their fine-grained resolution and saving them on a buffer, which allowed for keeping a low-memory profile while performing well on some incremental benchmarks. Liu *et al.* (2020) presented the use of an adaptive exemplar sampling for selecting replay instances and proposed different ways of applying the attention mechanism within the feature distillation procedure as a strategy to hinder forgetting. They evaluated their approach on various benchmarks and diverse scenarios in which the incremental data did not share the same domain as the base classes. Yang *et al.* (2021) proposed the use of a pre-trained language model to constrain the topology of the feature space within the model and capture the nuances of semantic relations associated with each class name. Their solution was meant to be used for open-world object detection. Still, it could also deal with incremental detection by using a replay buffer with prototypes for each class to prevent forgetting old categories.

Parameter Isolation

Parameter isolation solutions work by the assumption that the knowledge to perform tasks after training is purely within the model's weights. To prevent the current task's knowledge from vanishing in future updates, the model could have its parameters frozen and architecture modified to account for the new learning experiences (RUSU *et al.*, 2016).

Li *et al.* (2018) introduced a simple strategy for dealing with forgetting based on "mining" important network parameters and freezing them. For each task, they sorted the weight parameters by their magnitude and stored their values and positions in a memory buffer so that when training for the next task, the parameters would be reset to their original values. Zhang *et al.* (2021) proposed a compositional architecture based on the mixture of compact expert detectors. They trained a YOLO-V3 network using a sparse mechanism for each detection task and then applied the pruning technique suggested by Liu *et al.* (2017) to eliminate unimportant channels and residual blocks. For selecting which expert to forward the inputs to, they used a ResNet-50 classifier as the "oracle". Both strategies presented interesting results since the final model was able to keep a low memory footprint and little to no forgetting of old classes. Yet, their proposed solutions were only evaluated in limited scenarios (e.g., only three incremental tasks), making it difficult to compare to other techniques.

Pseudo-Labels

Considering that several issues of the CIOD setting are caused by the lack of annotations in new tasks that account for the old "already-known" classes, generating "fake-labels" based on the current knowledge of the model about these old classes proved to be a valuable alternative.

Guan *et al.* (2018) showed that when the base classes instances are also present in the images of the incremental categories, self-labeling using the own model could be a good enough strategy for dealing with forgetting. Liu *et al.* (2020) identified that pseudo-labels are an essential step when one wants to regularize the weight of a network with EWC. Moreover, they also introduced a novel Huber regularization loss for constraining the gradients of each parameter based on their relevance to the old classes. Yang *et al.* (2020) presented the use of pseudo-labels on the new classes images along with the application of general feature and output distillation and the learning of a residual model to compensate for the discrepancies between the teacher and student networks. Joseph *et al.* (2021) suggested the application of self-labeling to identify potential unknown objects on an image for open-world object detection. To prevent forgetting, they save a replay buffer with class prototypes and apply contrastive clustering in the feature space so that new classes can be added sequentially.

External Data

Also, to compensate for the possible differences in class distribution between the old and new tasks, some strategies considered using proper external data instead of storing samples or generating fake labels for the new task‘s images.

Li *et al.* (2019) used a one-stage detector (RetinaNet) and not only distilled the knowledge of outputs and intermediate features but also idealized a way to automatically collect and annotate new data from search engines such as the Google Image Search tool. These images were then used during the incremental training and testing schemes for improved performance. Zhang *et al.* (2020) proposed the independent training of one-stage networks on the base and new classes and the further transfer of their specific knowledge to a new separate network via knowledge distillation using an external unlabeled dataset. Dong *et al.* (2021) explored the scenario of non co-occurrence of old classes in new task images. They proposed a blind sampling strategy to select samples from large labeled in-the-wild datasets (e.g., COCO). To prevent forgetting, they designed a distillation strategy based on the remodeled output of the detection head, RoI masks on the image-level, and heatmaps on the instance level.

Meta-Learning

As described previously, Meta-Learning can be used to contain the model’s forgetting by framing it as a meta-level objective to the network. For this purpose, Kj *et al.* (2021) produced a hybrid strategy that relied on knowledge distillation, replay, and meta-learning to avoid forgetting. Along with the use of knowledge distillation on the outputs and backbone features, they used the gradient conditioning technique proposed by Flennerhag *et al.* (2019) to regularize the weight updates on some layers of the detector RoI head. This technique allowed fast adaptation to new tasks by fine-tuning the model with data from new classes and a few samples of old ones stored on a replay buffer.

Regarding the backbones used, the ResNet50 pre-trained on ImageNet-1k was the most used feature extractor for two-stage detectors and even some one-stage solutions (e.g., RetinaNet, Centernet and FCOS). Strategies that involved the YOLO architecture used mostly variations of the Darknet backbone. Considering that the ResNet50 was the backbone of the first work in the field, we believe that most posterior works followed this structure for making fair comparisons. In all papers, the weights of the backbone were not kept fixed even during the incremental learning steps to allow the fine-tuning of the feature extractor.

Several works used two-stage detectors and expanded the RPN network to account for the changes in the notion of “objectness”. Interestingly, the strategies focused on one-stage detectors did not propose specific mechanisms to prevent the drift in the background representation. These works seemed to prefer the use of one-stage solutions solely by their performance on the initial benchmark and improved inference speed (LI *et al.*, 2019; PENG *et al.*, 2021).

Nearly all methods were implemented in Pytorch, while only three claimed to use other frameworks, such as Tensorflow (SHMELKOV; SCHMID; ALAHARI, 2017; ZHOU *et al.*, 2020) or MatConvNet (LI *et al.*, 2018). In addition, some papers open-sourced their implementations, but we noticed a consistent lack of documentation in their repositories, which difficult the reproducibility and standardization of results (see Section 3.3 for a discussion on the lack of standards). For a new practitioner in CIOD, we point the readers to the efforts made by Joseph *et al.* (2021) and Kj *et al.* (2021) to share their Pytorch implementations[2,3] built on top of Detectron2 (WU *et al.*, 2019b) as starting points. For a reader more versed in Tensorflow, and who would be willing to analyze the field's "base" work, the original implementation of Shmelkov, Schmid and Alahari (2017) is also publicly available[4].

3.2.3.4 RQ4: What is the current state-of-the-art with respect to performance ?

For performing an investigation on which strategies have worked better for CIOD, we need to find common ground among them. Yet, there are no standards for frameworks, architecture backbones, and training regimes concerning paper reimplementations. Some papers tried to replicate the number of iterations, learning rates, and procedures used by Shmelkov, Schmid and Alahari (2017), but there is a clear difference in the obtained results that can be seen mainly for the joint training cases that used the same architecture (PENG; ZHAO; LOVELL, 2020; HAO *et al.*, 2019). In this way, it is difficult to state that some results are better than others because of the proposed policies and not due to the better selection of hyperparameters, which has been shown previously to highly influence generalization in the CL setting (MIRZADEH *et al.*, 2020). Therefore, using a consistent evaluation procedure is essential for identifying the most promising directions in the field.

Tables 3, 4 and 5, present the results of each paper that was evaluated on the PASCAL VOC 2007 and MS COCO 2014 following the main benchmarks described in Section 3.2.3.1 for when multiple and singles classes are added sequentially. The metrics proposed in Section 3.2.3.2 are used for evaluating the real impact of each strategy according to their upper bound. Because our metrics also need access to the *mAP* of each class for both the incremental and joint-training models, some previously discussed works have a † symbol that indicates that the paper only provided the mean *mAP* value for the old and new classes in groups for each setting.

In Table 3, by looking at the final *mAP* and the values of the upper-bound ratios for all incremental scenarios, it is possible to conclude that for the VOC benchmark, as more classes are added at once, the more complex the task becomes for the detectors. Strategies based on pseudo-labels and replay demonstrated consistent results. In contrast, pure knowledge distillation-based techniques struggled more and had an average plasticity deficit of more than 10%, which might be an indication that this type of regularization needs to be adjusted carefully to not harm the

[2] https://github.com/JosephKJ/iOD
[3] https://github.com/JosephKJ/OWOD
[4] https://github.com/kshmelkov/incremental_detectors

Table 3 – VOC 2007 results for one or multiple classes added at once

	VOC 2007 Incremental (1-19 + 20)			
Paper	Final mAP	$\Omega_{all} \uparrow$	RSD (%) $\downarrow$	RPD (%) $\downarrow$
Shmelkov, Schmid and Alahari (2017) (ILOD)	68.40	0.980	1.90	21.11
Li *et al.* (2018) (MMN)	77.50	0.991	1.09	-4.24
Li *et al.* (2019) (RILOD)	65.00	0.870	10.93	48.67
Peng, Zhao and Lovell (2020) (Faster ILOD)†	68.56	0.972	0.60	44.27
Zhou *et al.* (2020)†	69.60	0.991	-0.45	24.82
Zhang *et al.* (2020) (DMC)	70.80	0.948	4.80	12.33
Yang *et al.* (2020)	72.13	0.977	0.93	29.14
Shieh *et al.* (2020)	68.90	0.941	3.41	53.56
Ramakrishnan *et al.* (2020) (RKT)	67.20	0.984	1.00	14.29
Peng *et al.* (2021) (SID)†	68.30	0.954	4.61	4.61
Joseph *et al.* (2021) (ORE)	68.89	0.977	1.66	14.51
Yang *et al.* (2021)†	69.82	0.990	0.41	11.64
Yang, Zhou and Wang (2021)	69.70	0.973	2.11	12.17
Kj *et al.* (2021) (Meta-ILOD)	70.20	0.934	5.82	21.74
Dong *et al.* (2021)†	72.20	0.999	-1.38	29.88
	VOC 2007 Incremental (1-15 + 16-20)			
Paper	Final mAP	$\Omega_{all} \uparrow$	RSD (%) $\downarrow$	RPD (%) $\downarrow$
Shmelkov, Schmid and Alahari (2017) (ILOD)	65.90	0.944	3.60	12.33
Liu *et al.* (2020) (IncDet)†	70.40	0.954	0.44	12.12
Peng, Zhao and Lovell (2020) (Faster ILOD)†	67.94	0.963	-3.60	25.44
Yang *et al.* (2020)	69.71	0.944	1.92	17.75
Peng *et al.* (2021) (SID)†	62.20	0.869	13.13	13.13
Joseph *et al.* (2021) (ORE)	68.51	0.972	0.44	10.71
Yang *et al.* (2021)†	69.93	0.992	-3.55	13.93
Yang, Zhou and Wang (2021)	66.50	0.929	5.56	11.92
Kj *et al.* (2021) (Meta-ILOD)	67.80	0.902	6.58	20.62
Dong *et al.* (2021)†	65.30	0.903	2.49	31.67
	VOC 2007 Incremental (1-10 + 11-20)			
Paper	Final mAP	$\Omega_{all} \uparrow$	RSD (%) $\downarrow$	RPD (%) $\downarrow$
Shmelkov, Schmid and Alahari (2017) (ILOD)	63.10	0.904	7.66	11.42
Guan *et al.* (2018)	68.80	0.922	11.06	4.68
Chen, Yu and Chen (2019)†	33.50	0.474	47.05	69.02
Li *et al.* (2019) (RILOD)	67.90	0.909	10.42	7.67
Liu *et al.* (2020) (IncDet)†	70.80	0.959	4.52	1.18
Peng, Zhao and Lovell (2020) (Faster ILOD)†	62.16	0.881	-4.79	28.50
Zhou *et al.* (2020)†	61.80	0.880	9.16	14.89
Zhang *et al.* (2020) (DMC)	68.30	0.914	7.63	11.29
Yang *et al.* (2020)	66.21	0.897	5.98	14.74
Shieh *et al.* (2020)	65.50	0.895	8.78	14.04
Ramakrishnan *et al.* (2020) (RKT)	63.10	0.924	1.25	13.85
Peng *et al.* (2021) (SID)†	59.80	0.835	16.48	16.48
Joseph *et al.* (2021) (ORE)	64.58	0.916	14.76	2.01
Yang *et al.* (2021)†	64.96	0.921	14.86	0.89
Yang, Zhou and Wang (2021)	66.10	0.923	7.29	8.02
Kj *et al.* (2021) (Meta-ILOD)	66.30	0.882	8.51	15.07
Dong *et al.* (2021)†	59.90	0.828	20.33	13.97

learning of new categories. Nevertheless, for the settings with 5 and 10 classes added at once, although the initial baseline from Shmelkov, Schmid and Alahari (2017) presented a final *mAP* often below its competitors, it also offered a good balance of stability and plasticity according to its upper-bound, which contributes to why this technique is still relevant for comparisons.

Probably due to the increased complexity when working with several online updates, there were not many solutions to the sequential setting compared to its counterpart scenario, as shown in Table 4. For when only five classes were being added sequentially, the parameter isolation strategy of Li *et al.* (2018) demonstrated outstanding performance in the final *mAP* and stability-plasticity metrics. Also, in their unique participation, the RODEO method from Acharya, Hayes and Kanan (2020) confirmed that replay is a suitable tool for dealing with consecutive one-class updates. Interestingly, the IncDet strategy from Liu *et al.* (2020) performed well on the setup of multiple groups being added sequentially but seemed to fail when learning single classes alone. This may be related to how strongly the regularization penalty was adjusted to prevent the parameters from deviating much from the previously known distribution. In general, although having the same number of classes, the final *mAP* was clearly lower in this setting when compared to adding multiple categories at once. This corroborates that the “tug-of-war” on the parameters is happening actively in each new class network update.

Table 4 – VOC 2007 results for one or a group of classes added sequentially

	VOC 2007 Incremental (1-15 + 16 + ... + 20)			
Paper	Final mAP	$\Omega_{all} \uparrow$	RSD (%) $\downarrow$	RPD (%) $\downarrow$
Shmelkov, Schmid and Alahari (2017) (ILOD)	62.40	0.894	6.89	22.56
Li *et al.* (2018) (MMN)	76.00	0.972	2.21	4.48
Liu *et al.* (2020) (IncDet)†	67.60	0.916	1.12	35.87
Yang *et al.* (2020)	59.62	0.807	7.98	56.45
Peng *et al.* (2021) (SID)†	48.90	0.683	31.70	31.70
Kj *et al.* (2021) (Meta-ILOD)	65.70	0.874	8.77	25.08
	VOC 2007 Incremental (1-10 + 11 + ... + 20)			
Paper	Final mAP	$\Omega_{all} \uparrow$	RSD (%) $\downarrow$	RPD (%) $\downarrow$
Chen, Yu and Chen (2019)†	33.50	0.474	47.05	69.02
Acharya, Hayes and Kanan (2020) (RODEO)†	63.72	0.887	13.78	8.95
Zhou *et al.* (2020)†	46.20	0.658	22.46	45.82
	VOC 2007 Incremental (1-5 + 6-10 + 11-15 + 16-20)			
Paper	Final mAP	$\Omega_{all} \uparrow$	RSD (%) $\downarrow$	RPD (%) $\downarrow$
Hao *et al.* (2019) (CIFRCN)	48.50	0.694	36.62	12.09
Liu *et al.* (2020) (IncDet)†	62.60	0.848	11.58	21.67
Yang *et al.* (2020)	49.05	0.664	38.50	3.00
Ramakrishnan *et al.* (2020) (RKT)	52.90	0.775	20.56	29.23
Peng *et al.* (2021) (SID)†	36.20	0.506	49.44	49.44
Yang, Zhou and Wang (2021)	27.66	0.386	66.30	44.77

Considering the results for the COCO incremental benchmark exhibited in Table 5, techniques based on Faster-RCNN and feature distillation presented decent results. Even though the number of classes is higher than in the VOC benchmark, the upper-bound ratio shows that as the network is updated all at once, the forgetting condition is not as strong as in the sequential update example. Beyond that, methods based on self-labeling demonstrated results that justify their effectiveness for dealing with scenarios where a substantial number of classes was already introduced.

Table 5 – MS COCO 2014 results for multiple classes being added at once

	MS COCO Incremental (1-40 + 41-80)		
Paper	mAP@.5	mAP@[.5, .95]	Ω_{all} [@.5] ↑
Shmelkov, Schmid and Alahari (2017) (ILOD)	37.40	21.30	0.982
Liu *et al.* (2020) (IncDet)†	49.30	29.70	0.978
Peng, Zhao and Lovell (2020) (Faster ILOD)†	40.10	20.64	0.939
Zhou *et al.* (2020)†	36.80	22.70	0.868
Yang *et al.* (2020)	43.75	24.23	0.882
Peng *et al.* (2021) (SID)†	41.60	25.20	0.885
Yang, Zhou and Wang (2021)	44.62	-	0.854
Kj *et al.* (2021) (Meta-ILOD)	40.50	23.80	0.794
Dong *et al.* (2021)†	40.90	22.50	0.893

Overall, it is clear that all methods suffered from some aspect of forgetting and were limited to the joint-training baseline in all benchmarks. It is important to mention that this might not always be the case and that some CL strategies may surpass this baseline, which has been the case for some based on parameter freezing mechanisms (YOON *et al.*, 2017). The RKT strategy proposed by Ramakrishnan *et al.* (2020) was the best knowledge distillation method on the benchmarks it participated; however, most of the pure distillation-based techniques presented low plasticity probably due to the constraints imposed on the original weights. The IncDet model, which involved EWC regularization and pseudo-labeling, showed the most consistent results in all evaluated benchmarks. Yet, some strategies based on parameter isolation and replay that did well in individual benchmarks, such as MMN and ORE, demonstrated that there is still room for exploring alternatives and possibly combining them.

Regarding architecture choice, although one-stage detectors have fast inference and show predictive performance on par with their two-stage counterpart, there is no evidence (and analysis) that one family of detectors can be more prone to forgetting or adaptability than the other. By considering the takeaways from Section 3.2.3.3, most CIOD methods were built on top of regular strategies already used for CL and performed decently for the evaluated benchmarks. Nonetheless, methods that presented both novelty and effectiveness, such as ORE (JOSEPH *et al.*, 2021) and the one from Yang *et al.* (2021), were in fact optimizing for open-world detection. This can be a fair indication that such a related field can be further explored in the context of incremental detection.

This review did not consider other desired characteristics for the CL desiderata, such as the low memory footprint, which usually tends to overthrow parameter isolation strategies and fast adaptability to new categories. The meta-learning hybrid method from Kj *et al.* (2021)

presented results slightly superior to other knowledge distillation techniques. Regardless, the method quickly adapted to new tasks using only 10 replay samples for each category during fine-tuning. In this way, considering the CL desiderata for object detection discussed in Chapter 2, meta-learning hybrid methods can play an interesting role in class-incremental scenarios and should be investigated more.

3.3 Trends and Research Directions

Considering the main takeaways of the performed systematic review, in this section, we briefly discuss some of the observed trends and possible research directions in the CIOD field.

Realistic CIOD benchmarks: Although CIOD strategies have been producing encouraging results with respect to the joint-training baseline, most of the benchmarks used to evaluate such techniques are adaptations of conceivable real scenarios. An agent that is interacting and learning from a visual stream of data continuously can face new classes, that were not known previously, as well as changes in data and even label distribution. Also, depending on how much computing is available, situations for learning from one pass through the data (i.e., continual online learning) might be needed and explored. This leaves room for researchers to elaborate and propose benchmarks that improve on the potential generality and applicability of CIOD.

Hybrid methods can prevent more forgetting: The best-performing solutions to the class-incremental problem in object detection involved a combination of techniques to avoid catastrophic forgetting. This outcome agrees with the findings from other computer vision tasks (QU *et al.*, 2021) and corroborates with the fact that even the brain has multiple ways to prevent subtle task interference (HASSABIS *et al.*, 2017). One key point common to most hybrid methods was the fine-tuning of new classes given the representation of old categories using pseudo-labels or replay samples. This fine-tuning resulted in better results but can require a large buffer of samples and, similarly, an extensive hyperparameter search which might prevent its application in the real world.

Knowledge Distillation may (or may not) be all you need: It is straightforward to notice from Table 2 that most proposed strategies in the CIOD field use knowledge distillation as their primary mechanism to mitigate the effects of catastrophic forgetting. Comparing the results of the selected papers on the PASCAL VOC 2007 and MS COCO incremental benchmarks considering the metrics that assess the stability-plasticity of solutions, the differences between a recently proposed distillation technique such as the one from Peng *et al.* (2021) and the first work of Shmelkov, Schmid and Alahari (2017) are subtle. This either means that researchers might have been overfitting their solutions to the benchmarks or that simple logits and bounding box distillation are a considerably strong baseline. We believe the latter to be a more reasonable explanation.

Self-supervised learning for robust representations: Self-supervised learning has shown promising results for various closely related tasks, including continual classification, few-shot object detection, and weakly supervised object detection (BAR *et al.*, 2021; HU *et al.*, 2022; HUANG *et al.*, 2021). Still, there seems to be a lack of papers that focus on the possible contributions to the field of continual object detection. Assuming the generalization power normally associated with self-supervised representations, their exploration in the context of CIOD can be expected.

Efforts on meeting the CL Desiderata for object detection: The majority of the currently published CL research is done focusing on improving the last 0.01% of performance, sometimes considering unrealistic scenarios (e.g., use of task labels at test time). However, for real-world focused applications, strategies should also contemplate practical implementation aspects such as the computational burden and frequency of updates for the model. For class-incremental detectors, the desiderata described in Chapter 2 give an intuition of the main aspects that future research could focus on in order to increase practical adoption for researchers in academia and industry.

Standardization of implementations for CIOD: Research in CIOD suffers from poor standardization and has not fully adopted the advent already developed by the CL community for reproducibility, such as the Avalanche and Continuum libraries (LOMONACO *et al.*, 2021; DOUILLARD; LESORT, 2021). Besides that, there is no standard implementation for most of the discussed solutions to leverage fair comparisons. Although some available implementations are provided using the Detectron2 framework or its old form (PENG; ZHAO; LOVELL, 2020; JOSEPH *et al.*, 2021; KJ *et al.*, 2021), the interpretation of the changes to the original framework that are needed to reach the same results is often difficult due to the abstract structure of the repository. One step towards improving on this issue is the open-sourcing of the code for the regular baselines evaluated when proposing a new benchmark. Ideally, the implementation should envision using a well-established framework specific to the field, where a better description of the differences and human-readable code can be maintained. Nevertheless, the metrics proposed in this paper are also available as a tool for performing honest comparisons between solutions for the same benchmark.

Overcoming the overestimation of results: As also found in a recent survey for few-shot object detection (HUANG *et al.*, 2021), most works evaluated on the VOC and COCO datasets are using their training and validation splits for fitting the models and the testing set for selecting hyperparameters. This can lead to an overestimation of results and generalization problems when selecting techniques for good performance in the real world. A straightforward fix would be to use the original train/val/test splits as indicated by the datasets organizers and not perform contradictory actions to favor the proposed methods. Yet, as most researchers are using this setup to report their performance on the current benchmarks, it is sadly expected that the follow-up papers still keep the same choice of splits. We believe researchers should be more careful when proposing and evaluating their strategies for new incremental benchmarks to ensure a not biased outcome.

3.4 Related fields

Some related computer vision tasks already involve components that deal with incremental object detection in their pipeline. This section discusses a few of them shortly to make it possible for the readers to connect with other fields that can inspire and contribute to the current research of continual object detectors.

Open-World Object Detection

The set of possible objects a detector can encounter at test time in the wild is limitless. For dealing with the unknown and adapting to it, the field of open-world object detection has emerged as a possible solution to unify the paradigms of open-set and open-world recognition to class-incremental learning with object detectors (MUNDT *et al.*, 2020; BENDALE; BOULT, 2015). The solutions in this category are usually the combination of a structure to detect out-of-distribution samples (i.e., unknown objects) and a specific module to allow learning from them in an incremental manner. By modeling the unknown, researchers believe it is possible to reduce the label conflict and therefore enable more autonomous detection pipelines (JOSEPH *et al.*, 2021).

Incremental Few-shot object detection

When learning incrementally in robotics applications, models can be required to learn from data streams with only a few batches and several unseen classes. This scenario makes it difficult to apply the traditional batch learning used with neural networks and therefore needs a particular solution. The field of Incremental Few-Shot Object Detection (iFSD) looks for fast adaption of a trained network in situations of a low-data regime for learning novel classes (PEREZ-RUA *et al.*, 2020). This scenario is naturally more difficult than the plain CIOD paradigm since it assumes no large dataset is provided. Arguably, the current results on their benchmarks show a trend to focus more on the adaption to new classes than on avoiding the forgetting of old ones (LI *et al.*, 2021). This might indicate that research should be directed first at how to solve a less complicated problem (i.e., class-incremental object detection with large batches), which can give hints on how to move forward to more complex scenarios.

Continual Semantic Segmentation

The field of Continual Semantic Segmentation deals with the same difficulties of continual object detection (e.g., background label conflict) but at the pixel level. Most of the current solutions that have excelled in the field involve the techniques described in this review, such as pseudo-labeling (DOUILLARD *et al.*, 2021) and knowledge distillation (MICHIELI; ZANUTTIGH, 2019; CERMELLI *et al.*, 2020). Its application has a direct impact on real-world robotics navigation and should always be looked at closely by CIOD researchers for insights.

Zero-shot Object Detection

The Zero-Shot Object Detection paradigm consists of learning to detect new categories that are not present in the training set by using non-visual features that describe them (BANSAL *et al.*, 2018). Specifically, a pre-trained language model was originally used to model the semantics associated with the class labels. These relations were then used to guide the learning and inference of new unseen classes by a detector. The method proposed by Yang *et al.* (2021), which was described in Section 3.2.3.3, combined a zero-shot strategy with exemplar replay and showed decent results not only for open-world recognition (their primary goal) but also for some CIOD benchmarks. This might be an indication that innovations can be appropriately adapted and shared among these fields.

3.5 Final Considerations

This systematic review investigated how continual learning solutions have been applied to object detection tasks covering the topic‘s most explored benchmarks, metrics, and strategies.

For the literature review, we analyzed the reported performance of the leading papers in two popular benchmarks for the class-incremental scenario with the lens of a new metric explicitly proposed to look at how well a detector adapts and maintains its internal knowledge. We found out that even though most of the current research appeals to the single use of regularization-based techniques, specifically knowledge distillation, the methods that presented the best overall results on the evaluated benchmarks usually combine such techniques with replay, self-labeling, and meta-learning.

Finally, we discussed some of the main trends in the field, pitfalls and how researchers may avoid them, and a few related tasks that can inspire the proposal of new methods and possible future research intersections.

CHAPTER

4

EXEMPLAR REPLAY FOR CONTINUAL OBJECT DETECTION

In this chapter, we investigate various methods for selecting exemplars to be used in replay-based solutions for COD.

4.1 Introduction

Within the field of machine learning, CL has become a cornerstone for real-world applications, allowing models to adapt and learn from new data incrementally without forgetting previously learned knowledge. Among the various strategies for CL, replay has become the most widely-used approach due to its fast setup and guaranteed performance against catastrophic forgetting, even though it is not as explored as its counterparts in academia (ROLNICK *et al.*, 2019; PELLEGRINI *et al.*, 2020; PELLEGRINI *et al.*, 2022). In replay-based CL, a small subset of representative samples, known as exemplars, are stored with their annotations in a memory buffer and periodically revisited to minimize forgetting. This enables the model to refresh its "memory" of previous tasks, thus maintaining performance on earlier learned objectives while simultaneously adapting to new tasks (RATCLIFF, 1990).

Despite the prevalent use of replay in practical applications, there remains an open question regarding the optimal employment of replay for specific fields such as COD. When selecting replay samples, object detection poses distinct objectives compared to classification tasks since the rehearsal can take into account specific detection aspects, such as the unbalanced nature of the task and the different number of object instances in each image. Consequently, the exemplar selection and replay strategies that work well for classification might not be sufficient or the best fit when directly applicable to object detection. The absence of clear guidelines for replay in COD highlights the need for tailored solutions and research to adequately address this gap.

In this chapter, we investigate the use of different strategies to populate replay buffers in order to optimize the final performance of a COD task that involves class-incremental learning. By considering the unique challenges of COD, we aim to shed light on the best practices for replay in this context, offering guidance and methodologies that can be generalized to real-world applications.

4.2 Related Work

Different strategies have been traditionally employed for replay-based CL in classification tasks, such as the use of a random or class-balanced sample picker (HAYES *et al.*, 2021). Yet, there are more elaborated strategies, such as selecting samples that are closer or farther from the mean-of-features of each class (REBUFFI *et al.*, 2017) or populating the buffer by checking whether the current sample produces a higher or lower loss (HAYES; KANAN, 2021), which can serve as a proxy of how informative it is and therefore its importance.

In the realm of analyzing different ways to populate a replay buffer, a recent investigation on the use of replay for language learning tasks indicated that sampling from the global data distribution provides the best results for text classification, while a method that provides a balanced memory composition per task performs better for question answering (ARAUJO *et al.*, 2022).

For image classification benchmarks, Buzzega *et al.* (2021) proposed different simple tricks by patching naive experience replay. The tricks included using a reservoir strategy, a class-balanced and loss-aware sampler, and known strategies such as bias correction and exponential LR decay (WU *et al.*, 2019a; LI; ARORA, 2019). Their results showed that simple strategies can be effective when there is room for replay in vision tasks.

A few works were identified and described previously in Chapter 3 which investigated replay solutions for COD. In summary, these studies explored the idea of using replay combined with other CL strategies highlighting the gains of the final approach, and did not focus on the effect that replay alone would have in their context. For populating the buffers, most of them depended on using reservoir buffers fulfilled randomly either differing by the final buffer size or adding a compression mechanism to the stored samples (HAO; FU; JIANG, 2019; SHIEH *et al.*, 2020; ACHARYA; HAYES; KANAN, 2020).

More recently for COD, Liu *et al.* (2023) proposed to calibrate each new task distribution by populating a replay buffer with samples that reduce the Kullback Leibler (KL) divergence between the original category distribution and the replay subset. They reported results using a transformer-based architecture and several other components to mitigate forgetting and credited part of the final performance gain to the ability to preserve the initial category distribution across tasks.

4.3 Methodology

Based on the current literature for replay-based CL and existing solutions for COD, we propose the investigation of efficient strategies for populating a replay buffer in this context. For the following experiments, we adopt a task-balanced reservoir buffer filled sequentially with data from the previous task. We consider a real-world constrained scenario in which the buffer size is limited to 10% of the number of samples of the initial task. The following strategies are investigated:

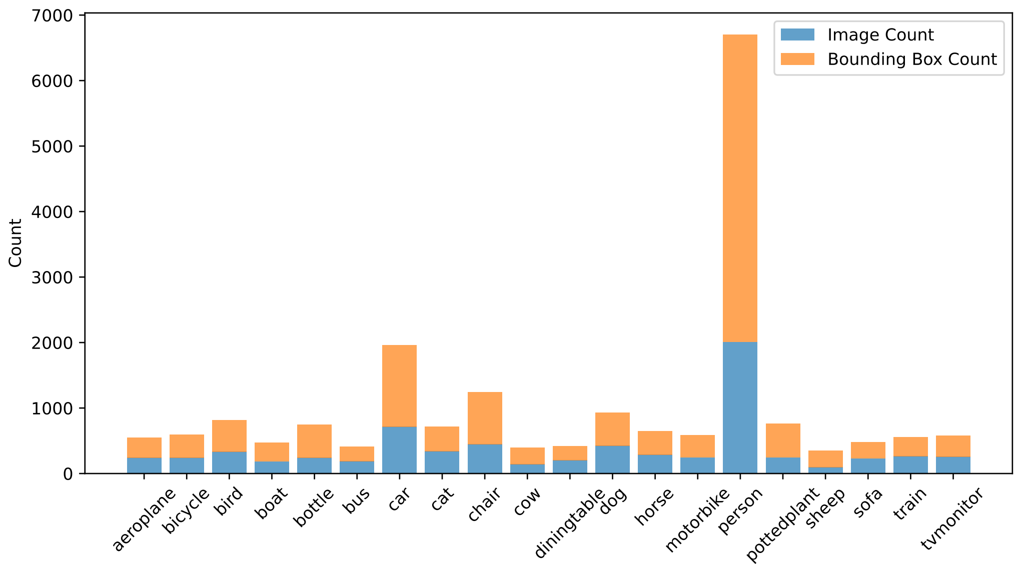
- **Random Buffer:** The buffer is filled by samples that are drawn randomly from the pool of samples of the previous task. This is the standard baseline for replay-based CL.
- **Class-Balanced Buffer:** Buffer is initialized so at least N samples from each class for the previous task are present for the next training experience. N is defined by the final buffer size divided by the number of seen classes and is adjusted for every task.
- **Max-Instance Buffer:** Considering that each image can have several annotated objects, samples are selected based on the maximum amount of present object instances.
- **Max-Loss and Min-Loss Buffers (HAYES; KANAN, 2021):** Buffer is initiated with a set of samples chosen by ranking the ones that produced the maximum or minimum loss for the currently trained model.
- **Distribution Matcher Buffer (LIU *et al.*, 2023):** Buffer is populated with samples that reduce the KL divergence between the current task and the replay buffer distribution.

4.3.1 Evaluation Benchmark

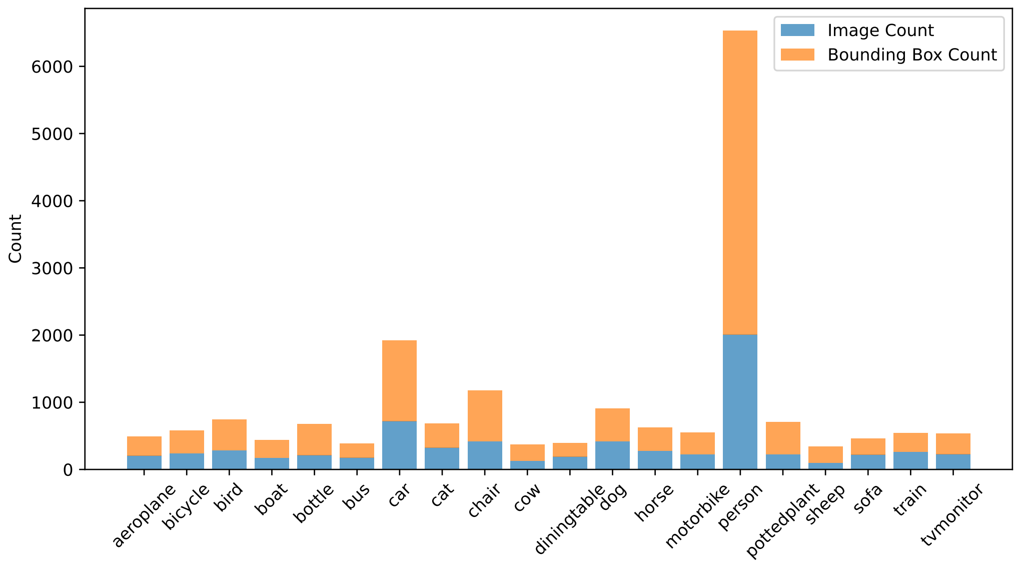
The benchmark for evaluation was the incremental version of the Pascal VOC dataset using the 2-step learning protocol used by the majority of works in the area (MENEZES *et al.*, 2023). We investigated the scenarios in which the last 10 classes and 5 classes are added at once or sequentially, as described in depth in the evaluated benchmarks presented in Chapter 3. Figures 15a and 15b show the distribution of images and boxes per class for the dataset. As for the evaluation metrics, we opted to report the final performance using the final *mAP*, as well as the upper-bound ratio (Ω_{all}), and the COD-specific metrics *RSD* and *RPD*, which were previously introduced in Chapter 3.

4.3.2 Implementation Details

Following the trend of other researchers who evaluated replay strategies, we opted to use the Faster-RCNN architecture without FPN Peng, Zhao and Lovell (2020), Joseph *et al.* (2021). The training hyperparameters followed the original setup used by Shmelkov, Schmid and Alahari



(a) Training distribution



(b) Testing distribution

Figure 15 – Distribution of images and bounding boxes per class.

(2017), where for the first task, the model is trained for 40k iterations with SGD and a LR of 0.01, and then, for the second task, the LR drops to 10% of its value. The model is then trained for 40k iterations when multiple classes are added at once or for 5k steps when individual classes are processed. The code for training the network and populating the buffer was written in Python and used the Pytorch Lightning (FALCON, 2019) framework.

To account for the inherent randomness in training neural network models and verifying the results' accuracy, we report results after averaging five runs with different seeds and conduct Wilcoxon signed-rank tests for each strategy pair.

4.4 Results

Table 8 summarizes the results obtained when training and evaluating each incremental scenario using the PASCAL VOC benchmark. The result for the *10+10 setting* indicates that the best-performing scenario happened when the class-balanced buffer was applied since it produced the least decrease in stability and deficits as well as a higher overall Ω_{all}. The second-best was the application of the distribution-matcher strategy, followed by the max-instance optimized buffer. However, when considering a 95% confidence interval for the chosen non-parametric test, Table 6 pointed out that there might not be a statistical difference for the result of the distribution-matcher, random, and max-loss strategies since their $p-value$ was above the threshold of 0.05. We hypothesize that, by drawing samples randomly from the initial task a sufficient number of times (i.e., 10% of the initial task sample size), the buffer will end up following the initial distribution and performing similarly.

Table 6 – P-values from the Wilcoxon signed-rank test for each pair of models in the 10+10 setting

	max_loss	min_loss	dist_matcher	max_instance	random	balanced
max_loss	-	-	-	-	-	-
min_loss	0.0	-	-	-	-	-
dist_matcher	0.7518	0.0	-	-	-	-
max_instance	0.0004	0.0	0.0015	-	-	-
random	0.1316	0.0	0.1290	0.0001	-	-
balanced	0.0089	0.0	0.0013	0.0	0.0	-

For the scenario in which 5 classes were added, the class-balanced buffer also presented the best results, followed by the randomly populated and distribution-matched buffer. By analyzing the $p-value$ shown for the model pairs in Table 7, we checked that there may not be a statistical difference among the results of the distribution-matched, max-instance, and max-loss strategies. Considering the nature of the data on the benchmark and its natural class unbalance shown by Figure 15a, there could be an overlap of the chosen images for the buffers since the increase of object instances in an image makes it more likely to be chosen for the max-instance strategy and can also impact on the loss of the model, which would make it be chosen for the max-loss optimized buffer.

Table 7 – P-values from the Wilcoxon signed-rank test for each pair of models in the 15+5 setting

	max_loss	min_loss	dist_matcher	max_instance	random	balanced
max_loss	-	-	-	-	-	-
min_loss	0.0153	-	-	-	-	-
dist_matcher	0.0689	0.0	-	-	-	-
max_instance	0.0944	0.0007	0.5958	-	-	-
random	0.0015	0.0	0.0144	0.001	-	-
balanced	0.0002	0.0	0.0	0.0	0.0023	-

For the scenario in which classes are added sequentially, the class-balanced buffer performed consistently better than the randomly populated buffer both when 5 and 10 classes were added individually. However, the model‘s general performance in this scenario was still lower than when all the class data was available for the incremental step. This bottleneck was also noted in the results of the other methods discussed in Chapter 3. In this situation, models usually struggle to find optimal solutions when not having knowledge of all objects present on the task. Additionally, by comparing the final *RSD* and *RPD* against the results of more complex techniques in the same benchmark, tailored specifically for several sequential updates (i.e., Table 4 in Chapter 3), we see that a well-thought replay strategy can be a simple and powerful baseline against most of the recently proposed strategies.

The final performance presented by the class-balanced buffer in all scenarios was consistently superior, even against more elaborated strategies based on matching the final distribution of the buffer. One possible factor for this finding is that, since we proposed an experiment with limited buffer size and the benchmark is naturally class-unbalanced, there is a higher chance of drawing samples with the more frequent classes on the distribution-matcher and random strategies. This helps the model maintain predictive performance on these objects in subsequent tasks, with the drawback of not producing a solution to remember long-tailed classes, which is something directly addressed by the class-balanced method. We also believe this to be one of the main reasons why the buffer that maximized for the number of object instances and the samples with the minimum loss did not produce competitive results, since they might have focused more likely on less diverse images with several instances of the same object, which can naturally produce a smaller loss.

In general, the *mAP* result for the scenario where 5 classes were added, on average, was higher than the counterpart with 10 classes. Some influential factors can be the use of a stronger base model, since it was trained with data from 15 classes instead of 10 classes, and the use of a larger replay buffer, since we considered a replay buffer size of 10% of the initial task‘s data.

Table 8 – Performance of FasterRCNN on the Incremental PASCAL VOC Benchmark for different replay strategies.

10 + 10 setting	aero	cycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	bike	person	plant	sheep	sofa	train	tv	**mAP**	$\Omega_{all} \uparrow$	**RSD (%)** $\downarrow$	**RPD (%)** $\downarrow$
Upper-bound	68.75	79.42	72.32	49.68	43.64	79.24	81.56	85.65	49.75	72.56	61.84	82.77	82.85	80.02	78.75	34.52	66.82	66.05	77.89	69.29	69.2	-	-	-
First 10	41.96	70.14	47.99	33.84	31.00	64.84	66.70	70.20	38.09	54.54	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	26.0	-	-	-
New 10	30.55	0.99	7.33	1.73	0.00	1.65	1.98	2.97	0.00	0.00	51.16	52.13	74.77	66.00	74.19	24.61	43.72	56.09	64.83	55.47	30.5	-	-	-
Fine Tunning + Class-Balanced Buffer	63.41	69.48	63.53	46.70	34.69	70.24	74.30	71.33	30.45	59.81	47.33	66.05	74.53	71.56	73.29	23.62	54.04	58.47	68.68	54.80	**58.8**	**0.85**	**14.26**	**15.68**
Fine Tunning + Random Buffer	57.94	65.86	60.25	42.02	29.63	63.93	74.13	71.37	29.30	51.09	42.83	66.99	75.10	70.76	72.86	22.96	47.79	55.88	65.58	53.29	56.0	0.81	19.81	18.33
Fine Tunning + Max Instance Buffer	58.46	64.21	58.45	43.88	33.98	70.70	76.56	47.55	38.69	58.14	43.97	54.97	76.70	71.07	73.06	24.86	52.91	57.25	67.75	55.06	56.4	0.82	19.08	17.81
Fine Tunning + Max-Loss Buffer	55.61	62.92	58.56	41.71	27.51	67.75	70.98	74.16	33.42	50.82	45.43	66.70	74.81	70.24	71.80	23.09	46.71	55.57	66.82	54.69	56.0	0.81	20.12	18.06
Fine Tunning + Min-Loss Buffer	57.44	54.97	57.48	27.03	22.59	59.44	66.87	40.84	17.19	33.37	45.90	54.76	76.47	71.15	73.96	24.98	46.03	55.97	66.26	53.49	50.3	0.73	35.47	19.06
Fine Tunning + Distibution Matcher Buffer	59.22	60.34	61.98	42.36	28.14	65.96	71.22	73.41	22.66	54.52	45.58	69.81	76.45	71.11	73.23	24.86	47.23	58.11	68.26	56.24	56.5	0.82	20.64	15.89
10 + 11 ... + 20 setting	aero	cycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	bike	person	plant	sheep	sofa	train	tv	**mAP**	$\Omega_{all} \uparrow$	**RSD (%)** $\downarrow$	**RPD (%)** $\downarrow$
Sequential Fine Tunning from class 11 to 20	2.43	0.00	0.00	0.00	0.00	0.00	0.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	8.36	49.47	3.1	0.04	98.19	92.96
Sequential Fine Tunning + Class-Balanced Buffer	61.74	73.45	58.38	46.55	35.74	70.55	72.07	70.39	32.27	55.90	15.17	38.49	50.86	57.54	47.12	22.34	50.61	55.74	49.15	54.36	**50.9**	**0.74**	15.25	**37.51**
Sequential Fine Tunning + Random Buffer	64.45	72.55	60.84	46.46	40.21	71.11	77.36	73.87	34.99	51.39	13.89	36.27	37.24	45.39	45.31	10.78	36.43	24.82	36.98	53.48	46.7	0.68	**12.92**	52.08
15 + 5 setting	aero	cycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	bike	person	plant	sheep	sofa	train	tv	**mAP**	$\Omega_{all} \uparrow$	**RSD (%)** $\downarrow$	**RPD (%)** $\downarrow$
Upper-bound	68.75	79.42	72.32	49.68	43.64	79.24	81.56	85.65	49.75	72.56	61.84	82.77	82.85	80.02	78.75	34.52	66.82	66.05	77.89	69.29	69.2	-	-	-
First 15	44.35	64.88	44.87	23.14	18.30	61.26	63.53	69.47	32.29	50.43	38.31	66.39	73.44	71.11	55.91	0.00	0.00	0.00	0.00	0.00	38.9	-	-	-
New 5	5.53	18.64	8.25	1.32	0.00	0.00	2.89	10.21	0.00	0.99	0.99	0.99	11.88	14.12	0.00	34.55	46.36	65.50	49.16	66.35	16.9	-	-	-
Fine Tunning + Class-Balanced Buffer	60.22	79.98	66.52	46.75	41.72	73.98	78.64	81.51	32.92	67.92	57.11	76.37	79.43	77.17	67.71	33.22	61.99	68.36	74.34	65.22	**64.6**	**0.93**	**7.79**	**3.31**
Fine Tunning + Random Buffer	60.83	78.67	62.84	42.08	33.49	63.97	78.39	79.03	33.01	59.94	55.25	74.82	78.74	77.23	69.42	33.22	60.08	65.64	73.33	65.07	62.3	0.90	11.67	4.98
Fine Tunning + Max Instance Buffer	49.37	79.45	64.67	41.41	41.87	68.85	79.97	49.49	34.85	54.78	58.82	45.43	79.15	75.93	72.30	30.92	56.07	66.62	73.77	65.37	59.5	0.86	16.62	6.31
Fine Tunning + Max-Loss Buffer	50.84	78.84	63.94	35.47	39.26	66.47	77.62	76.35	33.71	48.37	55.84	65.27	79.17	77.75	67.39	33.88	58.66	67.39	73.95	67.36	60.9	0.88	14.7	3.85
Fine Tunning + Min-Loss Buffer	57.51	68.63	65.44	37.02	26.35	67.26	76.12	73.34	18.05	63.65	26.67	67.02	77.56	69.53	49.58	31.83	56.76	65.53	70.40	65.44	56.7	0.82	21.7	7.12
Fine Tunning + Distibution Matcher Buffer	55.86	77.02	64.52	42.20	33.19	71.64	76.93	78.54	29.76	53.01	45.99	73.13	80.26	75.33	68.13	32.95	57.63	66.90	73.03	65.87	61.1	0.88	13.81	5.26
15 + 11 ... + 20 setting	aero	cycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	bike	person	plant	sheep	sofa	train	tv	**mAP**	$\Omega_{all} \uparrow$	**RSD (%)** $\downarrow$	**RPD (%)** $\downarrow$
Sequential Fine Tunning from class 16 to 20	6.19	13.22	1.89	1.88	0.99	3.62	10.62	2.92	0.00	2.90	0.99	0.99	4.86	9.27	0.99	6.68	7.86	12.29	11.67	51.94	7.6	0.11	97.1	64.81
Sequential Fine Tunning + Class-Balanced Buffer	64.26	80.83	67.41	45.13	42.45	71.17	79.95	82.75	39.87	65.28	59.51	75.43	80.46	77.26	73.96	20.81	55.17	51.43	63.92	45.09	**62.1**	**0.90**	6.08	**22.6**
Sequential Fine Tunning + Random Buffer	62.95	80.63	66.73	44.91	42.82	72.98	80.86	84.21	39.77	64.86	57.48	79.95	81.39	78.44	75.70	11.23	49.43	25.35	40.57	47.01	59.4	0.86	**5.31**	40.77

4.5 Final Considerations

This chapter elaborated on the most intuitive and beneficial ways to populate a replay buffer for COD. The obtained results indicated that, although randomly-populated buffers are still a strong baseline, for the Incremental VOC benchmark, class-balanced buffers present statistically superior performance. Additionally, the performance when classes were introduced sequentially showed that such buffers are a simple and efficient solution when compared to the results on the same benchmark discussed in Chapter 3. Another consideration against the use of more complex continual learning techniques is the computational time they may require for execution. A recent paper for CL in classification showed that several CL solutions are too computationally expensive for realistic budgeted deployment, which might also hold true for COD solutions (PRABHU *et al.*, 2023). In this sense, due to its simplicity and effectiveness, we argue that comparisons against replay strategies, especially using a class-balanced buffer, should always be performed when proposing strategies for COD.

It is also worth mentioning the importance of evaluating results through the lens of specific metrics, such as *RSD* and *RPD*. Incremental tasks may have a different number of new object classes and only looking at the final *mAP* and even Ω_{all} can be deceiving since the worsening of performance in one introduced class may not be enough to change the structure of these general metrics. Yet, by using criteria that individually scrutinize the difference in performance for the old and new task conditions, the strategies' final performance can be thoroughly evaluated.

CHAPTER

5

EFFICIENT PARAMETER MINING AND FREEZING FOR CONTINUAL OBJECT DETECTION

This chapter describes the investigation of efficient ways to mine, freeze, and penalize the change of important parameters of detector models when trained for COD tasks.

5.1 Introduction

Within computer vision, object detection is a fundamental task aiming at identifying and locating objects of interest within an image. Historically, two-stage detectors, comprising a region proposal network followed by a classification stage, were the norm, but they often suffer from increased complexity and slower run-time (ZOU *et al.*, 2019). The emergence of one-stage detectors, which combine these stages into a unified framework, has allowed for more efficient and often more accurate detection (TIAN *et al.*, 2020; LIN *et al.*, 2017b). In this context, incremental learning strategies for object detection can further complement one-stage detectors by facilitating the continuous adaptation of the model to new tasks or classes, making it highly suitable for real-world applications where the object landscape may change over time (LI *et al.*, 2019; HAQ *et al.*, 2021).

Recent works have concluded that catastrophic forgetting is enlarged when the magnitude of the calculated gradients becomes higher for accommodating the new knowledge (MIRZADEH *et al.*, 2021; HADSELL *et al.*, 2020). Since the new parameter values may deviate from the optimum place that was used to obtain the previous performance, the overall *mAP* metrics can decline. Traditionally in CL for classification, researchers have proposed to tackle this problem directly by applying regularization schemes, often preventing important neurons from updating or artificially aligning the gradients for each task. Such techniques have shown fair results at the cost of being computationally expensive since network parameters are mostly adjusted individually (KIRKPATRICK *et al.*, 2017; CHAUDHRY *et al.*, 2018).

To account for the changes and keep the detector aligned with their previous performances, most works in COD mitigate forgetting with regularization schemes based on complex knowledge distillation strategies and their combination with replay or the use of external data, as shown in the review presented in Chapter 3. However, we argue that the results presented by the solo work of Li *et al.* (2018) indicate that there is room to investigate further parameter-isolation schemes for COD. For such strategies, the most important neurons for a task are identified, and their changes are softened across learning updates in order to protect the knowledge from previous tasks.

In this chapter, we investigate different ways to identify and penalize the change in weights for sequential updates of a detector. By intelligently freezing these significant neurons, one might be able to avoid catastrophic forgetting and foster a more efficient and robust final model.

5.2 Related Work

The concept of using priors to identify the importance of the weights and protect them from updating is not new in CL. Kirkpatrick *et al.* (2017) proposed a regularization term on the loss function that penalizes the update of important parameters. These parameters are estimated by calculating the Fish information matrix for each weight, which considers the distance between the current weight values and the optimal weights obtained when optimizing for the previous task. Zenke, Poole and Ganguli (2017) similarly regularized the new learning experiences but kept an online estimate of the importance of each parameter. Both strategies compute the change needed for each individual parameter, which can be computationally challenging for large-scale detectors.

Also, on the verge of regularization, Li and Hoiem (2017) saved a copy of the model after training for each task and, when learning a new task, applied knowledge distillation on the outputs to make sure the current model could keep responses close to the ones produced in previous tasks. Such a strategy was adapted for COD in the work of Shmelkov, Schmid and Alahari (2017), which proposed to distill knowledge from the final logits and bounding box coordinates. Li *et al.* (2019) went further and introduced an additional distillation on intermediate features for the network. Both strategies have been used in several subsequent works in COD as strong baselines for performance comparison.

In CL for classification, Mallya and Lazebnik (2018) conceptualized PackNet, which used concepts of the neural pruning literature for applying an iterative parameter isolation strategy. It first trained a model for a task and pruned the lowest magnitude parameters, as they were seen as the least contributors to the model‘s performance. Then, the left parameters were fine-tuned on the initial task data and kept frozen across new learning updates. Such a strategy is usually able to mitigate forgetting, through the cost of lower plasticity when learning new tasks. Similarly, Li *et al.* (2018) proposed a strategy, in this chapter denoted as MMN, to "mine" (i.e., identify) important neurons for the incremental learning of object detectors. Their method involved ranking the weights of each layer in the original model and retaining (i.e., fixing the value of) the Top-K neurons to preserve the discriminative information of the original classes, leaving the other parameters free to be updated but not zeroed, as initially proposed by PackNet. The importance of each neuron is estimated by sorting them based on the absolute value of their weight. The authors evaluated this strategy with variations of the percentage of neurons to be frozen and found that a 75% value was ideal for a stability-plasticity balance within the model. Although simple, the final described performance was on par with the state-of-the-art of the time (SHMELKOV; SCHMID; ALAHARI, 2017).

The above parameter-isolation strategies for CL consider that the most important individual neurons will present the highest absolute weight values and must be kept unchanged when learning new tasks. This is a traditional network pruning concept and is commonly treated as a strong baseline (LECUN; DENKER; SOLLA, 1989; LI *et al.*, 2016). However, Neural Network Pruning strategies have evolved to also consider the filter and layer-wise dynamics. For that, the importance of a filter or layer can be obtained by analyzing the feature maps after the forward pass of a subset of the whole dataset. Then, they can be ranked and pruned based on criteria such as proximity to zero, variation inter samples, or information entropy (LIU; WU, 2019; LUO; WU, 2017; WANG *et al.*, 2021a). Even so, the available network capacity will be dependent on the number of involved tasks since important parameters are not allowed to change.

5.3 Methodology

From our review of the related work, we realized that there was room for investigating different ways to select the most important weights across learning experiences. Based on the recent neural pruning literature, we explore four different ways to identify important parameters to be kept intact across sequential updates. The following criteria are used to determine the importance of each network *layer* after forwarding a subset of images from the task data and analyzing the generated feature maps:

- **Highest mean of activation values**: Rank and select the layers with filters that produced the highest mean of activations.

$$I(layer_i) = \frac{1}{N}\sum_{k=1}^{N} F(x_k) \tag{5.1}$$

- **Highest median of activation values**: An alternative that considers the highest median of activations instead of the mean.

$$I(layer_i) = Med(F(x_k)) \tag{5.2}$$

- **Highest variance**: For this criterion, we consider that filters with higher standard deviation in the generated feature maps across diverse samples are more important and their layers should be kept unchanged.

$$I(layer_i) = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(F(x_k)-\mu)^2} \tag{5.3}$$

- **Highest information entropy**: Rank and select the layers based on the highest information entropy on their feature maps.

$$I(layer_i) = -\sum_{k=1}^{N} P(F(x_k))\log_2 P(F(x_k)) \tag{5.4}$$

where N is the number of images in the subset; $F(x_k)$ is the flattened feature map; Med is the median of the feature map activations; μ is mean of the feature map activations; P is the probability distribution of a feature map.

Additionally, in a separate investigation, we explore whether relaxing the fixed weight constraint proposed by MMN can allow the model to be more plastic while keeping decent performance on previous tasks. For that, we propose to simply adjust the changes to the mined task-specific parameters during the training step by multiplying the gradients calculated in the incremental step by a penalty value. By allowing them to adjust the important weights in a minimal way (i.e., with a penalty of 1% or 10%) across tasks, we hypothesize that the model will be able to circumvent capacity constraints and be more plastic.

For the proposed layer-mining criteria, we check which percentage (i.e., 25, 50, 75, 90) of frozen layers would give the best results. Figure 16 describes the proposed experimental pipeline.

5.3.1 Evaluation Benchmark

The incremental version of the Pascal VOC dataset with a 2-step learning protocol, also used in Chapter 4, was chosen as the evaluation benchmark. However, differently from the last chapter, we set the incremental step to either contain data from only the last class or the last 10 classes. For the metrics, we reported the final performance using the averaged *mAP*, as well as the upper-bound ratio Ω_{all}, and the COD-specific metrics *RSD* and *RPD*.

5.3.2 Implementation Details

We opted to explore the RetinaNet one-stage detector using a frozen ResNet50 with an unfrozen FPN backbone. The selected freezing criteria is therefore only applied to the neck (i.e.,

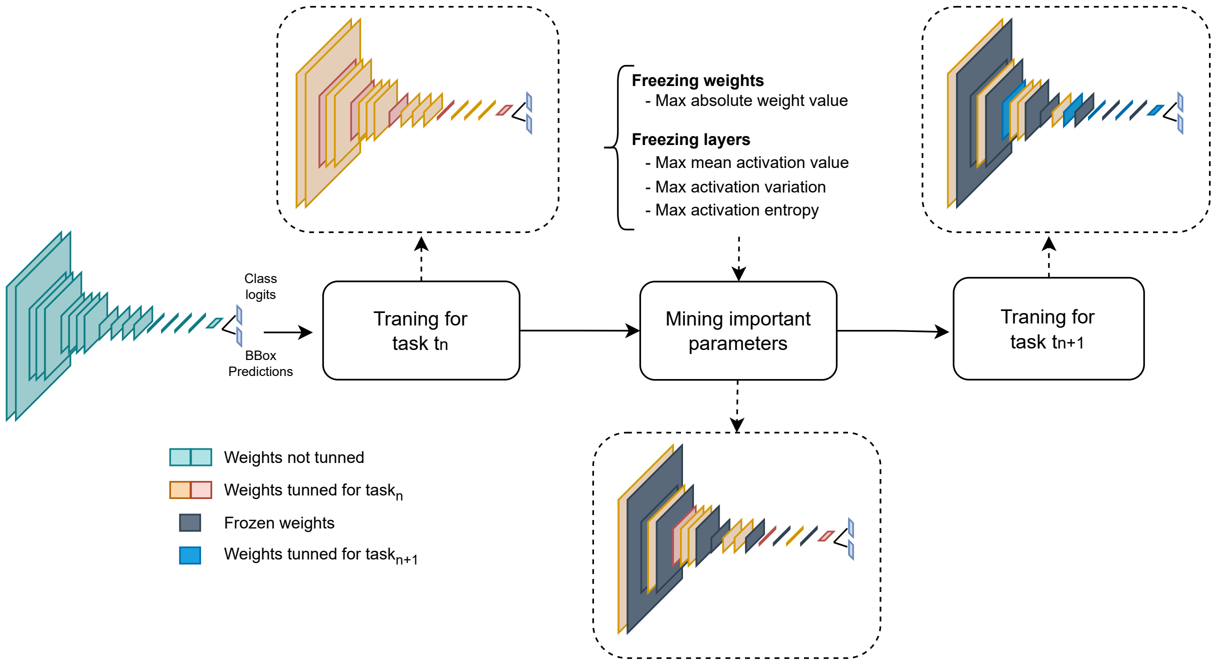

Freezing weights
- Max absolute weight value
Freezing layers
- Max mean activation value
- Max activation variation
- Max activation entropy
Class logits
BBox Predictions
Traning for task tn
Mining important parameters
Traning for task tn+1
Weights not tunned
Weights tunned for task$_n$
Frozen weights
Weights tunned for task$_{n+1}$

Figure 16 – Mining important parameters for efficient incremental updates.

FPN) and head of the model. The training settings are similar to the ones proposed by Shmelkov, Schmid and Alahari (2017) and also used in Chapter 4 in which the learning of the Pascal VOC benchmark is performed in two incremental steps. The model is trained with SGD for 40k steps with an LR of 0.01 for learning the first task and then with an LR of 0.001 for more 40k steps when 10 classes were added or 5k steps when only data from the last class is presented. The code for training the network was written in Python and used the MMDetection toolbox for orchestrating the detection benchmark and evaluation procedure (CHEN *et al.*, 2019). The main followed steps are depicted below in Algorithm 1.

As for the baselines, we consider the results reported on the work of Li *et al.* (2019) for the ILOD and RILOD strategies which also made use of the RetinaNet with ResNet50 as the backbone in a similar training setting. We also compare the results against our own implementation of the MMN strategy from Li *et al.* (2018) as well as the upper bound when all data is available for training the model. To account for the randomness associated with neural networks, we report the performance of each strategy after the averaging of three runs with different seeds.

5.4 Results

Table 9 describes the performance of each strategy for the $19+1$ scenario, while Table 10 reports the results for the $10+10$ alternative.

For the scenario in which the model was presented with data from only the last class, we noticed that the final *mAP* and Ω_{all} would heavily benefit models that were more stable than

Algorithm 1 – Incremental training with parameter mining and freezing for COD

```
 1: M: Model to be trained
 2: $Tasks$: List of learning experiences
 3: $S$: Type of mining strategy
 4: $L$: Percentage $L$ of frozen layers or parameters
 5: $P$: Percentage of gradient penalty
 6: $C$: Criteria for freezing the layers
 7: $N$: Percentage of samples from $Task_i$ to be used for calculating freezing metrics
 8: $i \leftarrow 0$
 9: for $i$ in range(length($Tasks$)) do:
10:     Train model $M$ with data from $Task_i$
11:     if $S = gradient\_mining$ then
12:         Dump previous gradient hooks
13:         Attach a hook with the gradient penalty $P$ to the selected percentage $L$ of parameters
14:     end if
15:     if $S = layer\_freezing$ then
16:         Reset $requires\_grad$ of the parameters in each layer
17:         Freeze a percentage $L$ of the layers given the chosen criteria $C$ using statistics from
            the feature maps obtained after forwarding the $N$ selected samples
18:     end if
19:     Fine-tune in $Task_i$ for $1k$ steps to regularize parameters for the next learning experience
20:     $i \leftarrow i + 1$
21: end for
22: return $M$
```

Table 9 – Results when learning the last class (TV monitor)

19 + 1		aero	cycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	bike	person	plant	sheep	sofa	train	tv	**mAP**	$\Omega_{all} \uparrow$	**RSD** (%)↓	**RPD** (%) ↓
Upper-bound		73.5	80.6	77.4	61.2	62.2	79.9	83.4	86.7	47.6	78	68.1	85.1	83.7	82.8	79.1	42.5	75.7	64.9	79	76.2	73.4	-	-	-
First 19		77	83.5	77.7	65.1	63	78.1	83.6	88.5	55.2	79.7	71.3	85.8	85.2	83	80.2	44.1	75.2	69.7	81.4	0	71.4	-	-	-
New 1		48	61.2	27.6	18.1	8.1	58.7	53.4	17.1	0	45.9	18.2	31.9	59.9	62.2	9.1	3.4	42.9	0	50.3	63.8	34.0	-	-	-
ILOD		61.9	78.5	62.5	39.2	60.9	53.2	79.3	84.5	52.3	52.6	62.8	71.5	51.8	61.5	76.8	43.8	43.8	69.7	52.9	44.6	60.2	0.81	18.01	45.66
RILOD		69.7	78.3	70.2	46.4	59.5	69.3	79.7	79.9	52.7	69.8	57.4	75.8	69.1	69.8	76.4	43.2	68.5	70.9	53.7	40.4	65.0	0.87	10.90	51.28
MMN	25	71.8	78.8	66.5	48.5	48.6	73.4	78.8	77.1	9.1	76.5	52.3	74.7	82.4	76.3	62.3	21.5	65.9	20.9	68.2	45.6	60.0	0.82	17.06	41.70
	50	73.4	79	71.5	51	53.4	73.4	81.6	78.5	13.9	73.5	54.5	76.7	83.2	79.1	64	27.7	66.8	36.3	69.4	43	62.5	0.85	13.23	45.24
	75	74.8	79.3	72.9	54.9	54	73.9	82	85	25.4	77.2	60	81.8	83.5	80.2	70.1	35.9	68	49.7	67.8	39.3	65.8	**0.90**	**8.25**	**50.29**
	90	76.5	82.4	74.4	58.4	57.9	74.2	82.3	86.7	35.7	77.6	65.1	83.7	83.8	82.2	72.5	37	73.2	58.5	71.5	33.7	68.4	0.93	4.15	57.92
Gradient penalty of 1%	25	71.9	78.8	66.5	48.6	48.5	73.4	78.8	77.1	9.1	76.5	52.3	74.6	82.4	76.3	62.3	21.5	65.9	20.7	68	45.5	59.9	0.82	17.08	41.84
	50	73.3	79	71.4	51	53.3	73.4	81.6	78.4	13.8	73.5	54.4	76.7	83.2	79	64	27.4	66.8	34.7	69.3	43	62.4	0.85	13.43	45.24
	75	75	79.3	72.9	54.9	54	73.8	82	84.9	25.3	77.2	59.8	81.8	83.5	80.1	70.1	35.8	67.9	49.3	67.8	39.4	65.7	**0.90**	**8.32**	**50.15**
	90	76	82.1	74.4	57.3	57.3	74.1	82.1	85.9	34	77.4	63.4	82.9	83.4	82	72.1	37.1	72.4	57.1	70.5	34.3	67.8	0.92	5.01	57.10
Gradient penalty of 10%	25	71.8	78.6	66.5	48	48.5	73.4	78.8	77.1	9.1	76.5	52.2	74.1	82.4	76.2	62.2	21	65.6	19.9	68.2	45.4	59.8	0.81	17.31	41.97
	50	73.1	78.8	71.3	49.6	53.3	74.5	81.5	78.3	11.4	73.4	54	76.4	82.8	76.8	63.8	27	66.4	33.4	68.6	43.8	61.9	0.84	14.13	44.15
	75	73.9	79.2	72.9	53.5	54.2	73.4	81.8	79.6	22	76.9	58.4	81.6	83.3	79.8	69.3	33.6	67.4	47.2	67.4	39.4	64.7	0.88	9.75	50.15
	90	76.2	81.8	73.6	55.9	57	73.2	81.2	84.6	30.3	76.9	60.7	82.4	83.6	81.1	71.1	36.3	68.3	56	67	37.2	66.7	**0.91**	**6.76**	**53.15**
Freezing based on mean	25	75.1	78.8	71.6	57.3	54.3	75.3	81.1	78.6	27.5	77	60.4	80.8	82.5	79.6	70.5	32.5	72.3	57.3	74.1	31.3	65.9	0.90	7.52	61.19
	50	75.3	78.6	72	57.7	53.8	74.7	81	79	27	74.7	62.5	77.8	82.7	77.5	70.5	33.1	72	56.5	73.1	32.4	65.6	0.89	8.03	59.69
	75	76	79.5	73.2	58	57	75.8	81.6	84.4	27.3	77.3	64.8	82.1	82.7	80.4	71.5	36	72.7	57.4	74.8	25.2	66.9	0.91	5.66	69.50
	90	76.2	81.3	71.9	60.8	49.9	75.7	82.8	86.2	24.8	76.5	69.4	82	82.9	80.9	68.5	26.2	71.9	60.3	79.4	41.7	67.5	**0.92**	**6.01**	**47.02**
Freezing based on median	25	75.1	78.7	71.7	57.3	54.4	74.8	81.2	78.7	27.4	76.9	60.1	80.8	82.5	79.3	70.6	32.3	72.5	57.3	73.6	31.3	65.8	0.90	7.62	61.19
	50	75.3	78.8	72.3	57.7	56.7	74	81.6	79.4	26.5	76.9	63.1	81.8	82.6	78.9	70.8	34.7	72.8	56.2	72.9	24.4	65.9	0.90	7.06	70.59
	75	78	79.6	73.2	57.1	55.7	76.1	82.6	86.1	38.3	77.2	65.8	83.1	82.4	80.5	73.7	38.5	71.6	60.5	75.4	31.2	68.3	0.93	4.02	61.32
	90	77.4	82.1	72.7	61.3	50.3	77.2	82.9	85.8	28.8	76.4	69.5	82	82.8	81.2	68.5	27.5	71.7	60.4	79.1	39.6	67.9	**0.92**	**5.29**	**49.88**
Freezing based on std	25	75.1	78.9	71.6	57.3	54.3	75.3	81.1	78.6	27.5	77	60.4	80.8	82.5	77.4	70.5	32.4	72.3	57.3	74	31.5	65.8	0.90	7.68	60.92
	50	75.1	78.9	71.6	57.2	54.3	75.3	81.1	78.7	27.5	77	60.4	80.7	82.5	77.4	70.5	32.3	72.3	57.3	74	31.4	65.8	0.90	7.70	61.05
	75	75.7	79.1	72.9	57.1	56.4	75.2	81.4	79.3	25.2	77.4	61.5	81.6	82	79.5	70.6	33.7	72.9	56.1	74.5	27.9	66.0	0.90	7.12	65.82
	90	77.6	79.9	73.5	57.3	56.6	77.7	82.8	86.2	38.2	77.1	65.9	82.8	82.5	80.2	73.7	39	72.4	61.5	76	31.5	68.6	**0.94**	**3.62**	**60.92**
Freezing based on entropy	25	75.5	79.4	72.7	56.2	57.2	74.8	81.9	84.7	28.9	77.9	62	81.4	83.1	81.1	71.6	35.3	68.4	54.7	69	40.7	66.8	0.91	6.86	48.38
	50	76.8	81.6	72.5	57	52.2	74.7	83.2	78.3	22.2	73.8	63.7	78.1	81.3	80	70.7	25.3	71	45.4	74.4	57	66.0	**0.90**	**9.27**	**26.17**
	75	76.9	81.8	71.9	61.4	50.4	76	82.7	86	29.5	76	69.6	82.3	82.9	80.7	68.6	26.7	72.1	60.9	79.6	40.5	67.8	0.92	5.41	48.65
	90	77.4	81.9	72.3	61.4	50.2	76.3	82.9	85.7	30	76	69.6	82.2	82.5	81.2	68.5	27.4	72	60.7	79.4	38.2	67.8	0.92	5.29	51.79

plastic since there was a clear imbalance in the number of represented classes (i.e., 19 $\rightarrow$ 1) for the incremental step. With that in mind, we analyzed the results that better balanced the decrease in *RSD* and *RPD* since, by splitting the deficits in performance, it is clearer to understand the ability to forget and adapt in each model. By comparing the results of the application of gradient penalty with respect to freezing the neurons with the highest magnitude (i.e., MMN in Table 9), we see that allowing the extra plasticity did not produce broad effects in performance. However, when 90% of the weights were mined, the extra adjustments introduced by using 1% of the calculated gradients allowed the model to beat MMN. Regarding the results of layer-mining, freezing based on information entropy presented a better balance in *RSD* and *RPD*, even against more established techniques such as ILOD and RILOD. For most of the results, increasing the percentage of frozen layers gave a lower deficit in stability with the caveat of increasing the difference in *mAP* against the upper bound for the new learned class.

Overall, leaving a lower percentage of parameters frozen across updates for the methods that worked on individual neurons made their networks more adaptable. Yet, this hyperparameter for the layer-freezing methods did not greatly affect the learning of the new class but had a significant impact on the detection of classes that had been learned previously.

For the $10+10$ scenario, the final *mAP* and Ω_{all} became more relevant as there was an equal representation of classes for their calculations. Results for applying a penalty to the gradient of selected neurons showed a slightly superior performance compared to completely freezing them. This was especially true in all scenarios where a 10% penalty was applied. For this benchmark, freezing 25% of the layers based on information entropy yielded the best results, followed by using the median of the activations to the same percentage of frozen layers. However, the final *mAP* and Ω_{all} indicate that these simply arranged strategies might have a difficult time competing against traditional methods when processing a benchmark with more complexities. Nonetheless, they can still serve as a quick and strong baseline when compared to fine-tuning and MMN due to ease of implementation.

Overall for the $10+10$ scenario, all evaluated strategies produced comparable final in terms of *mAP* and Ω_{all}. Nevertheless, the best outcomes were observed when freezing or penalizing 50% or less of the parameters. Since most detectors based on deep neural networks are overparameterized and not optimized directly for sparse connections, freezing more than 50% of available parameters or layers might affect highly the network capacity for learning new objects. We believe this to be true mainly for learning new tasks with imbalanced category sets and objects that do not present visual similarities with the ones previously learned. The Incremental Pascal VOC benchmark not only presents an imbalanced occurrence of each category (Table 15 in Chapter 4) but also a considerable semantic difference for the labels of the two tasks, with the first having more instances from outdoor environments and the second focusing on instances from indoor scenes. This can be further investigated by exploring task-relatedness as a way to define the parameters that determine how layer-freezing should take place between updates.

Interestingly, as also shown in the final evaluation remarks of the PackNet strategy for classification, the final performance of the incremental model can be weakened since it only uses a fraction of the entire parameter set to learn new tasks (DELANGE *et al.*, 2021).

Table 10 – Results when learning the last 10 classes

10 + 10		aero	cycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	bike	person	plant	sheep	sofa	train	tv	**mAP**	$\Omega_{all}\uparrow$	**RSD** (%) ↓	**RPD** (%) ↓
Upper-bound		73.5	80.6	77.4	61.2	62.2	79.9	83.4	86.7	47.6	78	68.1	85.1	83.7	82.8	79.1	42.5	75.7	64.9	79	76.2	73.4	-	-	-
First 10		79.2	85.6	76.5	66.7	65.9	78.9	85.2	86.6	60.2	84.7	0	0	0	0	0	0	0	0	0	0	38.5	-	-	-
New 10		0	0	0	0	0	0	0	0	0	0	74.6	85.7	86.1	79.9	79.8	43.9	76.3	68.5	80.5	76.3	37.6	-	-	-
ILOD		67.1	64.1	45.7	40.9	52.2	66.5	83.4	75.3	46.4	59.4	64.1	74.8	77.1	67.1	63.3	32.7	61.3	56.8	73.7	67.3	62.0	0.84	17.65	13.48
RILOD		71.7	81.7	66.9	49.6	58	65.9	84.7	76.8	50.1	69.4	67	72.8	77.3	73.8	74.9	39.9	68.5	61.5	75.5	72.4	67.9	0.93	7.59	7.29
MMN	25	59.2	37.4	38.7	33.3	17.2	46.3	52.9	57.5	5.9	45.7	62.9	73.6	76	68.8	77.1	37.6	62.9	60.9	72.5	73.5	53.0	0.72	45.84	9.72
	50	65.0	42.7	43.4	37.6	19.8	53.1	58.5	58.5	6.0	46.0	59.4	72.6	73.1	69.5	75.5	35.7	60.0	59.2	69.2	71.7	53.8	**0.73**	**40.89**	**12.44**
	75	61.5	40.3	49.0	35.8	19.5	48.0	54.8	52.3	10.5	44.0	62.5	71.0	74.1	68.4	75.6	36.2	59.6	61.3	69.6	70.7	53.2	0.73	42.91	12.00
	90	67.2	24.9	56	39.9	31.2	59.1	62.2	64.6	6.5	53.4	34.1	53.5	35.2	63.1	72.1	27.5	30	45.3	61.9	62.9	47.5	0.65	36.18	34.27
Gradient penalty of 1%	25	59.2	37.4	38.5	33.3	17.1	46.1	52.8	57.6	5.9	45.8	62.9	73.5	76.1	68.6	77.1	37.4	62.9	61	72.6	73.5	53.0	0.72	45.90	9.74
	50	64.9	43.9	43.3	37.2	19.3	53.1	58.4	58.4	5.6	46.0	59.3	72.7	73.1	69.6	75.6	35.8	60.2	59.2	69.4	71.8	53.8	**0.73**	**40.91**	**12.34**
	75	63.6	41.0	49.9	36.7	19.6	48.4	57.0	53.0	10.5	43.9	61.9	71.5	74.3	67.9	75.4	35.8	59.5	61.1	69.4	70.4	53.5	0.73	41.84	12.23
	90	67.2	25.1	55.2	41	30.1	58.9	62.2	63.9	5	52.9	38.2	55	44.5	64.9	72.5	28.6	35	47.7	62.6	64.4	48.7	0.66	36.66	30.49
Gradient penalty of 10%	25	59	36.8	36.5	33	16.5	46	52.7	56.8	5.8	45.8	63.1	73.7	76.5	68.6	77.1	37.9	63.2	61.1	73	73.3	52.8	0.72	46.55	9.48
	50	67.2	44	43.5	38	20.4	51.8	60.8	60.5	4.7	46.5	59.1	72.7	73.2	68.9	75.6	34.7	59.6	59	69.8	71	54.1	**0.74**	**39.94**	**12.74**
	75	66.5	44.1	50.8	37.0	19.5	52.1	57.2	56.1	8.3	46.2	60.4	70.2	73.0	68.7	75.4	35.4	59.3	58.7	69.3	70.9	53.9	0.73	39.93	13.08
	90	67.6	25.8	50.6	39.5	24.9	57.2	61.5	58.5	4.7	47.6	57.2	68.1	69.8	70.7	75.3	34.0	55.1	57.7	68.3	69.3	53.2	0.72	39.88	15.24
Freezing based on mean	25	63	48.4	57.3	36.1	19.9	57.1	49.8	66	7.7	45	54	64	64	70.4	72.1	33.9	49.7	58.6	62.1	66.6	52.3	0.71	38.18	19.31
	50	63.4	48.6	58	39.1	19	57.4	50	66.2	8.4	44.3	53.8	63.3	63.8	70.3	72.2	33.2	49.8	58.5	61.6	67.1	52.4	**0.71**	**37.63**	**19.56**
	75	58.8	49.1	55.6	41.1	17.5	58.1	43.5	67.5	11	43.3	47	66	54.3	70	70.2	32.4	47.4	58.8	51	67.5	50.5	0.69	38.84	23.51
	90	54.2	49.7	51.2	39.8	23.9	60.1	44.1	70.7	14.2	46.6	24.1	57.9	46.7	63.5	59.3	28.8	42	58.4	43.8	59.4	46.9	0.64	37.61	34.51
Freezing based on median	25	60.9	48.3	57.8	34.3	23	57.3	43.8	65.7	10.4	46.2	55.1	65.2	67.7	71.3	72.8	33.9	52.8	59.3	65	68.3	53.0	**0.72**	**38.54**	**17.13**
	50	58.5	48.8	55.4	41.5	18.7	58.4	43.8	70.5	11	41.9	53.7	66.8	54.2	71.2	71.8	35.1	49.4	59.6	52.6	68.7	51.6	0.70	38.43	20.99
	75	54.6	48.9	52.7	38.4	24.6	59.3	44.1	70.9	14.1	47.2	29.4	58.7	49.5	63.6	60.4	29	42.8	58.6	45.8	59.9	47.6	0.65	37.57	32.62
	90	53.6	42.4	51.9	38	23.8	60.1	44.1	71.3	14.4	47.5	28	58.7	49	64.7	60.1	25.4	42.3	58.4	46.8	59.7	47.0	0.64	38.62	33.25
Freezing based on std	25	62.7	48.5	57.4	36.2	19.6	57.1	49.8	66.1	7.6	45.2	54.1	64.1	64	70.2	72.2	33.9	49.8	58.4	62.1	66.4	52.3	**0.71**	**38.20**	**19.34**
	50	62.6	48.4	56.8	38.5	19.2	57.8	50	65.9	7	45.1	52.9	63.8	63.7	70.2	71.8	32.8	49.9	57.7	60.7	66.4	52.1	0.71	38.05	20.06
	75	62.1	47.3	57.8	38.8	19.5	58.2	50.1	65.3	8.5	44.6	53.4	62.7	64	69.9	71.5	31.7	51.1	57.1	60.8	65.1	52.0	0.71	37.93	20.41
	90	57.2	40.8	55	29.8	11.5	57.3	44.2	65.5	10.8	41.7	39.6	58.9	55.3	62.2	68.9	33.3	55.2	60	54.4	64.1	48.3	0.66	43.16	25.24
Freezing based on entropy	25	68.3	42.3	49.8	42.1	15.3	53.3	60.8	60.9	4.8	51.4	49.9	71.4	72.4	71	75.5	36.2	53.5	57.5	70.4	70.2	53.9	**0.73**	**38.36**	**14.87**
	50	60.8	34.1	48.2	30.1	32	51.8	42.2	56.9	14.9	45.3	55.7	63	67.5	66.5	73	32.5	46.9	58.8	62.3	67.4	50.5	0.69	42.82	19.56
	75	61.2	31.9	49.4	32.8	29.2	55.7	46.5	57.4	10.6	47.7	55.8	66.6	65.4	64.5	71.8	30.8	45.7	57.7	63.8	66.4	50.5	0.69	41.99	20.25
	90	54.6	53.6	63.8	46.0	24.4	55.9	53.4	69.4	20.0	51.6	31.4	53.7	49.1	59.2	40.0	7.5	31.0	55.0	41.1	34.8	44.8	0.61	32.43	45.58

However, this tradeoff is necessary to ensure stable performance in the tasks that were initially learned. Considering the necessity for quick adaptation in constrained environments, having a hyperparameter to adjust the plasticity of the model can be used as a feature to preserve the performance in previous scenarios and slightly adjust the network to the new circumstances. This feature can be especially beneficial when new updates with mixed data (i.e., old and new samples) are expected in the future.

5.5 Final Considerations

In this chapter, we discussed different ways to mitigate forgetting when learning new object detection tasks by using simple criteria to freeze layers and adjust how important parameters should be updated. We found that mining and freezing layers based on feature map statistics, particularly on their information entropy, yielded better results than freezing individual neurons when updating the network with data from a single class. However, when introducing data from several classes, the simple arrangements brought by the layer-freezing strategy were not as successful. The layer-freezing strategy's performance was on par with mining individual neurons, but not on par with more traditional and complex knowledge-distillation methods such as ILOD and RILOD. Additionally, results also showed that applying individual penalties to the gradients of important neurons did not significantly differ from the possibility of freezing them.

As a future line of work, it may be beneficial to explore fine-grained freezing solutions that involve mining and freezing individual convolutional filters based on their internal statistics. Hybrid techniques that balance learning with the use of experience replay could also be proposed to prevent forgetting and adapt more quickly to new scenarios. Furthermore, it would be useful to investigate measures of task-relatedness as a means of defining the freezing coefficients among sequential updates.

CHAPTER

6

CONTINUAL OBJECT DETECTION IN REAL-WORLD SETTINGS

This chapter reports two practical applications regarding Continual Object Detection in which we evaluated strategies and engineering solutions. The first section presents our approaches for the 3rd Track (Continual Instance Object Detection) of the 3rd CLVISION Challenge at CVPR 2022. Using a combination of knowledge distillation and balanced replay, our team secured the third position out of all participants in the challenge. The second section elaborates on the first incremental benchmark and evaluation scenario for COD using aerial inspection data.

6.1 Introduction

When training object detection models for real-world applications, large and specific datasets are often required. When a new dataset arrives, in cases of unseen classes or gradually changing data patterns, incremental training and adaptation become of vast importance. This is especially relevant for applications such as robotics, industry inspections, and autonomous vehicles since these scenarios may have restrictions on data privacy and available computational power for retraining (SHAHEEN *et al.*, 2022).

Despite its importance, most of the existing solutions for continual learning in computer vision have focused on classification scenarios, while object detection in an incremental manner remains a more challenging task. This is also reflected in the number of competitions that were proposed recently for computer vision conferences and workshops, and the number of participants for each of them (LOMONACO *et al.*, 2022; BAE *et al.*, 2020; PELLEGRINI *et al.*, 2022).

As discussed in the systematic review presented in Chapter 3, incremental learning strategies for object detection are often divided into class-incremental and domain-incremental instances. However, in the realm of the real world, where possibilities are endless, models may need to deal with situations that combine both challenges. The following sections explore solutions for COD in two real-world tasks in which the model needs to incorporate the knowledge of new object classes and adjust to changes in existing classes without requiring complete retraining on all the data.

6.2 Continual Egocentric Perception

6.2.1 Context

Several robotics and augmented reality applications require that perception models (e.g., computer vision models for visual understanding) run successfully in a first-person perspective, also known as egocentric (GRAUMAN *et al.*, 2022). Such a scenario is challenging since most of the available visual data in the world comes from datasets in the traditional form of a “spectator” or third-person view. The captured footage of an agent engaging with its environment often features unusual poses and varying illumination, making low-level visual tasks and contextual interpretation of human actions more challenging. Additionally, an agent can experience new scenarios and situations daily, making this context an interesting test-bed for models that can deal with such fast-changing scenarios.

To encourage research on the frontiers of egocentric perception, the 3rd CLVISION Challenge at CVPR 2022 proposed the exploration of a massive-scale egocentric dataset for detecting objects from a continual learning perspective. In this context, the challenge presented three possible tracks involving continual learning for object recognition, with Track 3 focusing on continual instance-level object detection. The 34 registered teams for this track were required to handle five learning experiences containing a total of 1111 different objects, with the final solutions constrained to use only pre-trained methods on the ImageNet-1K, COCO, or LVIS datasets. Other constraints related to the final solution included:

1. **Max model size**: 70M parameters.
2. **Max replay buffer size**: 5000 samples.
3. The model needed to finish its **training and evaluation under 24 hours** on the reference server [1].
4. **No test time training or augmentation**.
5. The solution **must not** use information regarding the category of instances nor the video ID at test time.

[1] The reference server was an AMD EPYC 7282, 128 GB RAM @ 2666 MHz with SSD and an Nvidia Quadro RTX 5000.

6.2.2 Ego4D dataset

The dataset used for this challenge is an adaptation of the Ego4D dataset (GRAUMAN *et al.*, 2022), released by Meta, for the incremental setting called EgoObjects. The dataset features first-person videos of people handling objects in their daily lives. The videos were broken into frames and split into training and testing sets containing no data leakages. Through their API, which was made available by the competition organizers, the training set could also be easily split into training and validation. An example of the images present on the dataset is shown in Figure 17.

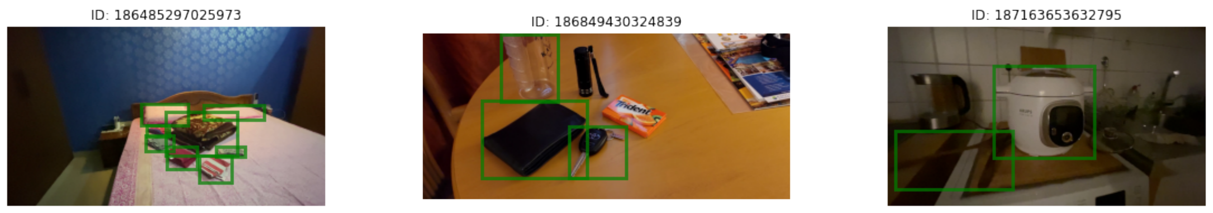


Figure 17 – Samples from the EgoObjects dataset.

The challenging aspect of incremental learning in this context is that each experience can present new classes as well as training samples from previous learning experiences. This mix of class and domain-incremental scenarios was proposed as a way to represent how continual learning would have to be explored in a real-world situation.

6.2.3 Methodology

Considering the aforementioned data and constraints, in this section, we describe the main components of our proposed solution for the challenge. The Avalanche library, a Pytorch-based framework for CL (LOMONACO *et al.*, 2021), was used to load the challenge data and structure the CL strategy.

6.2.3.1 Architecture and training settings

For the neural network architecture, we chose the Fully Convolutional One-Stage Object Detector (TIAN *et al.*, 2020) using a ResNet50 with FPN as the backbone from torchvision[2]. The model was pre-trained on the COCO train2017 dataset (LIN *et al.*, 2014) and had only its head changed to be able to detect all the 1111 possible objects.

For the optimizer and learning rate across experiences, we applied SGD with a 0.05 LR and momentum of 0.9. The LR scheduler was set to be linear with a warmup of 1000 iterations, as in the general template given for the competition. The scheduler was applied only to the first epoch of the first experience and kept the LR stable across the whole training execution. Although this scheduler was chosen for our final solution, some preliminary tests using a StepLR

[2] https://pytorch.org/vision/0.12/models.html

starting at 0.01, decreasing by 50% every 10,000 steps, and restarting at each experience also showed a decent performance. Yet, we did not have time to explore diverse solutions based on this last setup.

Other general training settings were:

- **Number of Epochs:** 5.
- **Image size**: images were limited to 800 on the largest size and 600 on the shortest.
- **Training batch size**: 4.
- **Validation batch size**: 16.
- **Transforms**: Random Horizontal flip with 50% of probability.

For evaluation, the average of the mAP scores across experiences was used, here denoted as Experience Average Precision (EAP). Since the initial metric calculates the mean of the AP across class instances in every single experience, class-imbalanced experiences would not hurt the final performance as much.

6.2.3.2 Balanced Experience Replay

Considering the maximum buffer size and that the number of images for each object instance was highly unbalanced (e.g., some with more than 100 instances, others with 15), we proposed the use of a balanced replay buffer.

After the first experience, the replay buffer was initialized so that at least N samples from each class for the previous experience were present for the next task. In this case, N is the buffer size available for the task divided by the number of different object instances in it. The buffer size was defined by the maximum buffer size divided by the number of tasks seen so far.

Using such a strategy instead of an experience replay buffer based on reservoir sampling resulted in an increase from 34.6 to 40.8 on the final leaderboard metric (average mAP across experiences).

6.2.3.3 Knowledge Distillation from Features and Outputs

Following the basic distillation procedure for incremental object detection introduced by Shmelkov, Schmid and Alahari (2017) and the following advances proposed by Chen, Yu and Chen (2019), we regularized the learning of each new experience by distilling knowledge from a saved version of the model trained on the previous experiences. The distilled knowledge came from the L_2 loss, here named penalty, applied to the head (logits, bounding boxes, and "center-ness" of objects) and intermediate features (layer2.3.relu, layer3.3.relu, and layer4.2.relu) for both models as described by Equations 6.1 and 6.2.

$$L_{head} = \frac{1}{3}\sum_{j}\frac{1}{M}\sum_{i=1}^{M}||y_j^{teacher}(x_i) - y_j^{student}(x_i)||^2 \tag{6.1}$$

$$L_{feat} = \frac{1}{3}\sum_{k}\frac{1}{M}\sum_{i=1}^{M}||F_k^{teacher}(x_i) - F_k^{student}(x_i)||^2 \tag{6.2}$$

where y was the output of a head j, F were the feature activations from a layer k and x_i was a sample from a batch M for the current experience. The final penalty was calculated by adjusting the weight for the L_{feat}, as shown in Equation 6.3 since the losses were on different scales. We found that a λ of 10 was able to balance the contribution of the two terms.

$$L_{penalty} = \lambda\ L_{feat} + L_{head} \tag{6.3}$$

The final loss used to update the weights was the original L_{FCOS}, obtained when training the student detector, summed by the scaled penalty value as described in Equation 6.4.

$$Loss = L_{FCOS} + \alpha\ L_{penalty} \tag{6.4}$$

α was a parameter to calibrate the current model's stability-plasticity considering the previous one. Most distillation solutions in continual object detection use the value of 1 as reference (PENG; ZHAO; LOVELL, 2020; CHEN; YU; CHEN, 2019), but we found that the value of 0.5 had better performance for the final validation metric. This distillation setting resulted in an increase from 40.8 to 41.69 AP, as shown by our final performance on the leaderboard.

6.2.4 Results

Table 11 presents the final mAP after each experience and the final EAP for the top 3 submissions. Additionally, Table 12 presents a summary of the design choices of each ranked solution on the track.

Table 11 – mAP after each experience and final EAP

Team	E0	E1	E2	E3	E4	EAP
Tencent Youtu Lab (1st)	0.2330	0.3953	0.5459	0.7021	0.8560	0.5465
NUSA*STAR (2nd)	0.1505	0.3044	0.4550	0.6079	0.7537	0.4543
Our solution (3rd)	0.1474	0.2909	0.4232	0.5539	0.6691	0.4169

The winning solution used experience replay and knowledge distillation on the heads and features, along with a more recent and robust detector and backbone. The runner-up proposed a hybrid replay and architectural solution based on a multi-head FasterRCNN and Non-Maximum

Table 12 – Description of each final ranked solution

Team	Base Detector	Backbone	Pretraining Dataset	Replay	Distillation	AP	AP50
Tencent Youtu Lab (1st)	VarifocalNet	Res2Net101	COCO	Experience	X	54.7	61.2
NUSA*STAR (2nd)	Faster R-CNN	ResNet50	LVIS	Video		45.4	56.0
Our solution (3rd)	FCOS	ResNet50	COCO	Instance	X	41.7	51.1

Supression (NMS) for fusing the detected objects. Although our solution did not place as well as the winner, we managed to beat the 4th place by around 3-4 mAP points.

As discussed in the final competition report for all tracks, even though populating and selecting important samples for the replay buffer takes computational time that could be used for employing more complex solutions, a properly tuned replay strategy was crucial to prevent forgetting and obtain a good balance between stability and plasticity in all solutions for the detection tracks (PELLEGRINI *et al.*, 2022).

6.2.4.1 *Failed attempts to improve the results*

We evaluated several strategies to improve the incremental detector's final performance. Due to the considerable computational time required when training large detectors, most of the experiments were executed for 5 epochs to briefly check their effectiveness. Some of the attempts are described below:

- **Adam optimizer**: The optimizer presented decent initial performance but was less stable than SGD for higher LRs and thus had worse validation metrics for 5 epochs.
- **Learning Rate Finder + One Cycle Policy (SMITH, 2017)**: Strategy presented good learning performance but had lower validation metrics than plain SGD with LinearLR and warmup iterations.
- **Larger images**: Training with larger images (Max 1333 x Min 800) showed better validation results but impacted the total used memory and training time. Thus, we kept the smaller setup to comply with the evaluation server memory and time constraints.
- **More epochs**: Considering the increase in computational time brought by the distillation component, we limited the training to 5 epochs to comply with the constraints mentioned above. However, some initial tests pointed out that training for more than 5 epochs would not result in any considerable gains in performance. This might also be related to using a LinearLR that does not change across all the experiences.
- **Different Augmentations**: We applied some "stronger" augmentations, such as random distortions and IOU crops, but the final validation metrics were affected negatively.

It is worth noting that other exploratory experiments utilizing techniques from other chapters have been omitted due to our inability to duplicate the same conditions as the initial experiments. This was mainly because the test set for the competition was unavailable up to this manuscript‘s writing, and the evaluation server was not reachable.

6.3 Continual Detection for Aerial Inspection of Transmission Towers

6.3.1 Context

Preventive inspections of transmission towers are essential to ensure the safety of civilians and workers as well as evaluate the lifecycle of their components. This entails a thorough examination of their various parts, such as insulator strings, conductor cables, lightning rods, and spans, to detect any defects or irregularities that may pose risks.

These inspections are often performed by onsite specialists using binoculars or Unmanned Aerial Vehicles (UAVs), as shown in Figure 18, for first capturing video footage and then categorizing the health aspect of each component. Automating such tasks by the use of drones equipped with perception modules (i.e., object detection algorithms) has been largely approached in the industry setting (NIKOLIC *et al.*, 2013; SCHOFIELD; LORENZEN; EBEID, 2020).

Figure 18 – Real UAV flight during an inspection.

Since some automation pipelines depend on the detection being performed in real-time on the device, the inference speed, as well as the incremental aspect of updating the list of objects of interest, needs to be taken into account. For that, the use of models with slimmer architecture (e.g., one-stage detectors) equipped with CL strategies may be a reasonable solution for leveraging such applications.

6.3.2 Methodology

6.3.2.1 TAESA Transmission Towers Dataset

The detection of transmission towers and their components using aerial footage from drones is an essential step for executing inspection missions in transmission facilities (TAKAYA *et al.*, 2019). Besides the advantages in staff safety and the cost of the inspection, the use of UAVs for this task is also known to have a positive impact on the standardization of the acquisition process. However, there is a lack of successful reports of general applications in this field since it inherently involves several challenges related to acquiring training data, having to deal with large domain discrepancies (i.e., electric transmission towers can be located anywhere in a country), and the necessity to update the detector every time a new accessory or tower needs to be mapped.

To aid in the proposal of solutions for some of the listed issues, we introduce the TAESA Transmission Towers Dataset. It consists of aerial images from several drone inspections performed on energy transmission sites maintained by the TAESA company in Brazil. The full dataset has records from different transmission sites from four cities with different soil and vegetation conditions. In this way, the incremental benchmark was organized into four different learning tasks, each representing data from a specific transmission site, as illustrated by Figure 19.

Each task can have new classes that were not introduced before and new visuals for a previously introduced object, making it a challenging "data-incremental" benchmark. In addition, different from most artificial benchmarks, images were annotated by several people using a reference sheet of the possible classes that could be present. For that, the possibility of missing annotations and label conflict in posterior tasks was reduced. A summary of the dataset with respect to the number of images and objects, with their description, for each task can be seen in Tables 13 and 14.

Table 13 – ID for each class in the TAESA dataset.

Class Label	Description
0	Classic Tower
1	Insulator
2	Yoke Plate
3	Clamper
4	Ball Link
5	Anchoring Clamp
6	Guyed Tower
7	Support Tower
8	Anchor Tower

Table 14 – TAESA Dataset Summary.

			N° of Boxes per label									
Scenario	Set	N° of Images	0	1	2	3	4	5	6	7	8	Total Boxes
Task 1	Training	526	690	2228	482	119	381	528	-	-	-	4428
	Validation	67	78	245	55	16	29	49	-	-	-	472
	Testing	69	91	252	49	10	42	60	-	-	-	504
Task 2	Training	431	86	950	260	4	-	-	20	429	8	1757
	Validation	55	14	120	32	-	-	-	2	55	-	223
	Testing	55	2	120	29	1	-	-	3	55	-	210
Task 3	Training	308	5	726	269	39	-	-	303	-	4	1346
	Validation	39	3	92	31	5	-	-	36	-	-	167
	Testing	39	1	89	33	6	-	-	38	-	-	167
Task 4	Training	227	5	1242	357	-	770	83	-	-	234	2691
	Validation	28	2	165	50	-	98	12	-	-	29	356
	Testing	29	-	177	52	-	112	11	-	-	29	381

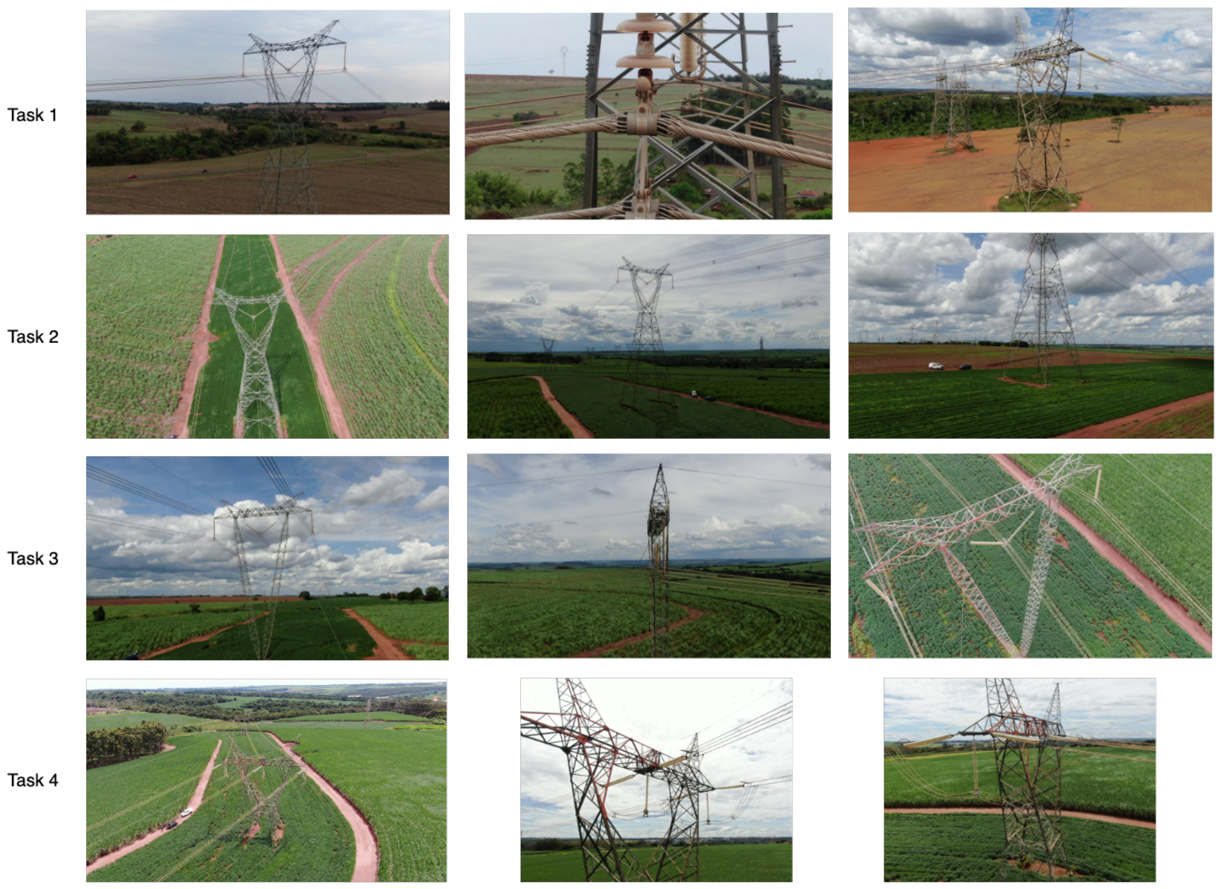


Figure 19 – Sample of images of each task for the TAESA Transmission Towers Dataset.

6.3.2.2 Implementation Details

For making evaluations on the proposed incremental benchmark, we use the same training setting and steps proposed in Chapter 5 in which a RetinaNet model with a ResNet50-FPN backbone is used for continual training, and different freezing and penalty strategies are tested. The model was trained for 40k steps for the initial task, which has more images and classes, and subsequently for 5k steps in each new task, using the same hyperparameters described in Chapter 5. As for the baseline, we compare it against the fine-tuning without any CL strategy, the use of individual neuron freezing as in the work of Li *et al.* (2018) denoted as MMN, task-balanced random reservoir replay with a buffer size of 10% of the first task‘s size, and the upper-bound defined by training with all the available images.

We report the results as the average of three runs with different seeds when learning each task sequentially. The performance is measured by the final *mAP*, with different thresholds, and *mAP*[.50] after learning all tasks, as well as with their upper-bound ratios Ω_{mAP} and $\Omega_{mAP[.50]}$. To account for a model's stability and plasticity, we have modified the existing *RSD* and *RPD* metrics to consider tasks instead of individual classes. In this evaluation scenario, *RSD* measures the performance deficit against the upper-bound in all tasks up to the last one, while *RPD* evaluates the performance deficit against the last learned task.

6.3.3 Results

Table 15 summarizes the results on the proposed benchmark with the green color highlighting metrics related to *mAP* and blue for $mAP_{[.50]}$. As the benchmark involves class-incremental and domain-incremental aspects, we noticed that when there is little drift in the appearance of previously known objects that show up in the new task images, these instances reinforce the “old knowledge” and can be considered as a small case of replay. This can be checked by the fact that the forgetting in the fine-tuning approach is “soft” when compared to other artificial benchmarks, such as Incremental Pascal VOC, in which classes that do not appear in further training sets are completely forgotten. Furthermore, the benchmark was organized in a way that minimized label conflicts, leading to less interference in the weights assigned to each class.

Applying a penalty to the gradients of important parameters improved the results of leaving them frozen (i.e. MMN) in all scenarios. The best results were seen when applying a 1% of the penalty to 50% or more of the important weights. Due to a slight imbalance between the number of available data and classes in each task and the fact that the first task had more learning steps, it was found that keeping most of the old weights unchanged, or slightly adjusting them to new tasks, proved to be effective for average performance. However, when checking the performance in the intermediate tasks (i.e., Tasks 2 and 3) and comparing them to the fine-tuning and upper-bound results, we see that forgetting still occurs, but to a lesser extent than in the other evaluated methods.

Selecting the most important layers based on information entropy was the most impartial in terms of the percentage of layers chosen, and generally yielded superior outcomes compared to other statistical measures. Yet, freezing 75% of the layers based on the mean of feature map activations seemed to produce the best results, achieving a good balance in the final Ω_{mAP} and $\Omega_{mAP[.50]}$, although it significantly impacted knowledge retention in intermediate tasks The other

Table 15 – Results for incremental training on the TAESA Benchmark

			Task 1		Task 2		Task 3		Task 4		Final Eval					
	%	Feature	mAP	mAP[.50]	mAP	mAP[.50]	mAP	mAP[.50]	mAP	mAP[.50]	**Average mAP**	**Average mAP [.50]**	$\Omega_{m}AP \uparrow$	$\Omega_{m}AP[.50] \uparrow$	$RSD_{mAP} \downarrow$	$RPD_{mAP} \downarrow$
Freeze	25	mean	43.7	67.9	5.6	13.5	13.3	24.1	35.1	60.8	24.4	41.6	0.55	0.60	51.18	28.22
		median	43.8	65.4	9.7	21	15.2	36.9	37.9	64.5	26.6	47.0	0.60	0.67	46.48	22.49
		std	41.7	62.5	10.5	21.6	19.3	32.9	38.6	64.9	27.5	45.5	0.62	0.65	44.28	21.06
		entropy	41.2	61.4	15.6	30.3	21	34.7	39.8	67.1	**29.4**	**48.4**	**0.66**	**0.69**	**39.33**	**18.61**
	50	mean	44.0	69.6	5.8	13.9	11.8	23.2	35	61	24.2	41.9	0.55	0.60	51.96	28.43
		median	43.3	64.7	10.5	22.5	14.8	26.3	37.2	62.6	26.5	44.0	0.60	0.63	46.52	23.93
		std	41.4	64.4	10.9	22.8	19.8	34.3	38.4	64.9	27.6	46.6	0.62	0.67	43.77	21.47
		entropy	41.0	61.8	16.6	31.5	22.2	37.8	39	65.9	**29.7**	**49.2**	**0.67**	**0.71**	**37.77**	**20.25**
	75	mean	47.9	71.4	3.5	9.8	12.4	24.1	31	55.3	**31.4**	**49.0**	**0.71**	**0.70**	50.28	36.61
		median	45.9	65.3	6.8	17.5	17.4	30.6	32.9	60	30.9	48.7	0.70	0.70	45.37	32.72
		std	44.1	63.2	10.8	24	19.3	32.5	34.4	62.1	30.5	48.7	0.69	0.70	42.14	29.65
		entropy	43.7	63.1	11.6	21.9	22.5	38.5	36.6	62.3	30.4	48.7	0.69	0.70	**39.33**	**25.15**
	90	mean	46.2	69.9	6.8	13.9	9.9	20.7	23.3	44.9	21.6	37.4	0.49	0.54	50.95	52.35
		median	45.4	68.8	8.6	22.8	15.8	29.9	25	48.5	23.7	42.5	0.53	0.61	45.62	48.88
		std	44.8	68.6	13.1	27.6	18.4	33.4	25.7	49.7	25.5	44.8	0.58	0.64	40.54	47.44
		entropy	45.6	67.0	13.9	28.5	19.5	33.8	28.4	53	**26.8**	**45.6**	**0.61**	**0.65**	**38.43**	**41.92**
Grad	25	0.1	44.2	67.8	7.5	16.6	20	34.5	37.2	64.4	**27.2**	**45.8**	**0.61**	**0.66**	**44.14**	23.93
		0.01	29.2	65.7	8.8	18	19.9	34.1	37.9	64.7	24.0	45.6	0.54	0.65	54.84	**22.49**
	50	0.1	45.7	69.7	9.7	21.4	18.8	32.6	35.2	61.7	27.4	46.4	0.62	0.67	42.16	28.02
		0.01	45.4	67.9	11.2	23.1	20	34.9	37.1	64.3	**28.4**	**47.5**	**0.64**	**0.68**	**40.28**	**24.13**
	75	0.1	47.5	70.6	9.7	23	18.5	31.6	31.5	57.7	26.8	45.7	0.61	0.66	40.97	35.58
		0.01	47.0	71.6	21.1	36.5	19.2	32.6	32.3	59.4	**29.9**	**50.0**	**0.67**	**0.72**	**31.96**	**33.95**
	90	0.1	48.7	72.9	15.6	31.1	17.7	32	28	53.1	27.5	47.3	0.62	0.68	36.09	**42.74**
		0.01	49.2	73.5	20.4	39.4	18	32.3	27.9	53.7	**28.9**	**49.7**	**0.65**	**0.71**	**31.69**	42.94
MMN	25	-	44.6	68.0	5.1	12.2	17.8	31.3	33.5	60	25.3	42.9	0.57	0.62	47.36	31.49
	50	-	47.3	69.7	4.2	10.1	17.4	31.7	31.5	58	25.1	42.4	0.57	0.61	46.33	35.58
	75	-	49.4	72.7	6.7	15.9	15.5	28.8	28.1	52.1	24.9	42.4	0.56	0.61	44.16	42.54
	90	-	48.6	72.0	10.4	18.6	14.2	26.8	13.8	32.5	21.7	37.5	0.49	0.54	42.97	71.78
Fine tuning	-	-	44.2	66.6	5.4	12.8	12	23.5	34.9	61.5	24.1	41.1	0.54	0.59	52.02	28.63
Experience Replay	-	-	46.7	71.3	21.5	37.8	24.9	40.6	42.5	71.9	33.9	55.4	0.77	0.80	27.40	13.09
Ground Truth	-	-	56.8	83.2	35.7	58.1	35.8	62.1	48.9	75.3	44.3	69.7	-	-	-	-

layer-freezing methods attained similar results, but with less forgetting in the intermediate tasks. This highlights the necessity to look at the big picture and not only specific metrics based on averages.

Although the full benchmark seemed challenging by having to deal with new classes and domains, the initial task's diverse and abundant data helped prepare the model to learn with small adjustments in new task scenarios. All evaluated strategies performed better than fine-tuning and MMN baselines but fell behind the results achieved through experience replay. For scenarios where saving samples is not feasible, a hybrid strategy involving parameter isolation and fake labeling may help reduce the gap in performance against replay methods. Nevertheless, when possible, combining these methods with parameter-isolation strategies can be seen as a promising direction for investigation.

6.4 Final Considerations

In this chapter, we described our strategies for real-world applications that involved COD in perception and an aerial inspection benchmark.

For the first part, as shown by our ablation studies, the catastrophic forgetting effects when training incrementally with the Ego4D dataset could be mitigated. For that, the best setting happened when using an experience replay buffer balanced by the number of tasks and different categories in each experience, along with a knowledge distillation component applied to the features and head outputs.

For the second part, we explored different freezing strategies and penalties for important parameters in order to be able to learn and adapt incrementally to new power transmission sites. The use of the entropy criteria for selecting which layers to freeze, instead of freezing individual weights, showed promising results even when only 25% of them were frozen across sequential updates. The obtained results were on par with a replay-based strategy, making layer-mining, as well as the proposal of hybrid approaches involving it, an interesting research topic for further exploration.

Considering the possible future directions for the ones who want to develop solutions to the same benchmarks, we believe that strategies that account for the label-conflict problem when they occur, such as self-labeling, and more advanced architectures and losses such as the VarifocalNet (ZHANG *et al.*, 2021) and Swin (LIU *et al.*, 2021) can be promising mainly if there are no or few initial constraints to the evaluated benchmark (e.g., number of parameters and training time). Additionally, to more strongly validate the effectiveness of layer-freezing for applications in COD, we should make comparisons against newer and more complex strategies taking into consideration other training perspectives such as the use of memory and computational time.

CHAPTER

7 CONCLUSION

In this final chapter, we present our concluding thoughts on this Ph.D. thesis and its contributions to the field of COD.

7.1 Final Considerations

Despite the fact that data availability has increased exponentially over the years, the time and processing power it takes to learn from such data can still be limited for several applications. This highlights the importance of the COD field in a world where machines require perception to plan and act in conjunction with humans. In relation to the objectives and hypotheses introduced in Chapter 1, some key considerations can be emphasized:

Hypothesis 1. *Metrics specifically tailored to highlight changes in the stability-plasticity of a model are more suited to class-incremental object detection than standard CL metrics.*

Chapters 3 and 4 touch on the necessity of more specific metrics to deal with COD scenarios. The proposed metrics introduced in Chapter 3 (*RSD* and *RPD*) give a simple way to compare methods regarding their ability to retain and acquire new knowledge through sequential updates. These metrics are particularly useful when there is an imbalance between the number of classes and available images for each incremental task with respect to the previous ones. This is because the average *mAP*, and consequently Ω_{mAP} generally used for CIOD, can mask the lack of ability to learn new classes or retain old knowledge by smoothing through the average result. Additionally, *RSD* and *RPD* can be adapted to represent the changes in stability and plasticity for scenarios with several incremental tasks, as shown in Chapter 6.

Based on our subjective analysis, the experiments conducted confirm our first hypothesis that metrics specifically tailored to highlight changes in the stability-plasticity (i.e., *RSD* and

RPD) are generally more suitable for evaluating COD results than the general average and forgetting metrics.

Hypothesis 2. *Class-balanced replay buffers are more effective for class-incremental object detection than using random buffers.*

In Chapter 4, we explored different ways to populate replay buffers for CIOD. Object detection poses inherent challenges, such as handling highly imbalanced tasks and diverse data. We found that the class-balanced replay buffer delivered the most consistent results across all evaluated scenarios for the Incremental Pascal VOC benchmark, demonstrating significant differences when compared to randomly populated buffers and other detection-optimized methods. Furthermore, in the first section of Chapter 6, we presented results that supported these findings for more practical applications of COD in the real world. With these scenarios in mind, we are confident that the given hypothesis holds true and recommend that COD researchers take it into account when evaluating their methods.

Hypothesis 3. *The use of a well-selected parameter mining and freezing strategy can enable deep neural network models to continually learn how to detect new objects while avoiding forgetting old ones.*

Two different strategies for identifying, freezing, and updating important weights during incremental learning steps were assessed in Chapter 5 and the second application reported in Chapter 6. The findings suggest that freezing layers based on their feature map activation statistics, and applying a gradient penalty to important weights instead of completely freezing them, can serve as strong baselines, particularly when dealing with tasks with fewer label conflicts and added classes. The obtained results outperformed the baseline where mined neurons were kept frozen, as proposed by Li *et al.* (2018), in both the $19+1$ Incremental VOC and TAESA benchmarks, but fell short of the use of experience replay and more complex regularization techniques in the $10+10$ scenario for the Incremental VOC benchmark. While we believe that these strategies confirm the hypothesis when compared to other competitive baselines, we also believe that there is still room for improvement and exploration, particularly when combining these strategies with other CL methods.

Overall, there are limitations regarding the coverage of our experimental claims since we opted to evaluate mostly in controlled scenarios. The CIOD-specific metrics rely on calculations using the raw class-wise or task-wise *mAP* values after detection and their upper-bound reference. While we used these metrics in Chapter 3 to compare methods that displayed their full results in the Incremental Pascal VOC benchmark and some in COCO, it is challenging to accurately compare the results of these metrics for papers reporting results in larger datasets like LVIS. This is because researchers usually report only the final average *mAP*. As we have stated many times in this thesis, it might be misleading to only report the final *mAP* or upper-bound ratio, and results should be interpreted with caution. To ensure fair comparisons, we suggest that researchers always evaluate their COD strategies with respect to plasticity and stability in a comparable way.

When developing new methods for CIOD, it is important to consider the use of class-balanced buffers as a starting point for comparison. However, it is crucial to keep in mind that these buffers rely heavily on the assumption that the model can store samples during learning updates. While storage may be a more cost-effective solution in some applications, there may be situations where rehearsing old data is not possible. In such cases, strategies that are based on parameter isolation and regularization may be the only viable option. Therefore, we suggest that experience replay for CIOD should always be considered when evaluating new strategies, but researchers should carefully assess when it is appropriate and develop strategies that are not solely dependent on it.

Limitations around parameter isolation can be related to the manual adjustment of a network's stability and plasticity abilities through the percentage of parameters/layers to be frozen, which can be useful in some situations but not practical in others. Furthermore, such simple techniques are a great choice against more computationally expensive strategies such as EWC or LWF, but may be not suitable for non-stationary tasks in which the importance of the weights and layers may change over time. To address that, the proposal of hybrid methods that deal with these scenarios can be considered for a more versatile solution.

In summary, for this thesis, we presented: a systematic review that helped organize and evaluate what had been done in the field of COD; an evaluation of how exemplar replay could be better utilized for the incremental learning of detectors; an exploration of different ways to mine, freeze and update important parameters of a network in order to deal with sequential learning; two indicative applications of how continual detection can and is used in the real world. We believe the individual works that formed this thesis are of great importance for future researchers who will build upon current solutions in the COD field.

Regarding future work and directions, we suggest that researchers should explore solutions beyond mainstream techniques. It is also important to consider approaches that have been successful in other fields, like open-vocabulary and open-world detection. Our investigations and the results we reported in Chapters 5 and 6 lead us to believe that simple techniques focused on model parameters and architectural changes should not be ignored, as they provide strong baselines for the COD task. However, we recognize the limitations of our work and suggest that future research should include large-scale detection benchmarks, such as the LVIS dataset(GUPTA; DOLLAR; GIRSHICK, 2019), to test new hypotheses in the context of COD. Besides that, applying self-supervision techniques in order to create object-agnostic representations, as already investigated in open-world object detection, can be a promising direction since it detaches the discrimination and localization aspect of the learning paradigm, which can be further explored by COD strategies.

7.2 Contributions

As direct contributions from this thesis to the research community, we name the following:

- The first review in COD through the paper entitled: “Continual Object Detection: A Review of Definitions, Strategies, and Challenges” published in Neural Networks in 2023.
- The proposal of two metrics (i.e., RSD and RPD) for the more thorough evaluation of COD pipelines.
- A solution that got 3rd place for Track 3 (Continual Instance Detection) on the CLVISION Challenge at CVPR 2022.
- The paper “Exemplar Replay Evaluation for Continual Object Detection” to be sent for review.
- The paper “Efficient Parameter Mining and Freezing for Continual Object Detection” to be sent for review.

Not directly related to this Ph.D. thesis, we had a few other contributions to academia and the Brazilian machine learning research field during the graduate program, such as:

- The writing of the book in Portuguese: “Ciência de Dados: Fundamentos e Aplicações” with Prof. André C. P. L. F. de Carvalho and prof. Robson Parmezan to be published by Editora Grupo Gen at the end of 2023.
- The paper accepted for publication at IEEE Access entitled “Sim-to-Real Transfer for Object Detection in Aerial Inspections of Transmission Towers” with researchers from the Eldorado Research Institute.
- The paper accepted for presentation at the “Open Innovation Week 2023”, entitled “A Framework for Multi-Rotor UAV Image Inspection of Transmission Towers” with researchers from the Eldorado Research Institute.
- The teaching of the course “Advanced Topics in Intelligent Systems: Deep Learning” in the Big Data and Intelligent Systems Specialization at SENAI Paraná in 2020.
- The teaching of the course “Introduction to AI with Computer Vision” within the national “SmartCities” project of the city Canaa dos Carajás in partnership with CEMEAI 2021.

- The preparation of the base material for the “Data pre-processing and preparation” and “Deep Learning” courses for the Master in Business Intelligence at SENAI Paraná in 2021.
- The participation in three Bachelor Thesis Committees from 2020-2023.
- The review of several papers for prestigious conferences and journals such as KDD, ECML, BRACIS, and Neural Networks from 2020-2023.
- The supervision of Davi Filetti and Pedro Conrado in their undergraduate research.
- The work as a teacher assistant for the “Machine Learning” and “Data Science Fundamentals” courses at the University of São Paulo in 2021 and 2022.
- The presentation of a workshop entitled “Artificial Intelligence and Neuroimaging” for the Medical League in Neurosurgery of Sergipe in 2020.
- The presentation of a tutorial at Python Nordeste entitled “Object Detection: From Zero to Hero” in 2022.
- The development of the machine learning backend of the NeuroKeypoint AR 2.0 app for allowing the precise inspection of brain lesions in real-time with a camera using pre-operative images, face registration, and augmented reality. The paper describing the approach is still in progress.

BIBLIOGRAPHY

ACHARYA, M.; HAYES, T. L.; KANAN, C. Rodeo: Replay for online object detection. **arXiv preprint arXiv:2008.06439**, 2020. Citations on pages 49, 52, 53, 58, and 66.

ACHARYA, M.; KANAN, C. 2nd place solution for soda10m challenge 2021–continual detection track. **arXiv preprint arXiv:2110.13064**, 2021. Citation on page 44.

AHN, H.; CHA, S.; LEE, D.; MOON, T. Uncertainty-based continual learning with adaptive regularization. **Advances in Neural Information Processing Systems**, v. 32, 2019. Citation on page 31.

ALAMMAR, J. **The illustrated transformer**. 2018. Available: <http://jalammar.github.io/illustrated-transformer/>. Citations on pages 11 and 30.

ALJUNDI, R. Continual learning in neural networks. **arXiv preprint arXiv:1910.02718**, 2019. Citation on page 31.

ALJUNDI, R.; KELCHTERMANS, K.; TUYTELAARS, T. Task-free continual learning. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2019. p. 11254–11263. Citation on page 32.

ALLEN-ZHU, Z.; LI, Y. What can resnet learn efficiently, going beyond kernels? **Advances in Neural Information Processing Systems**, v. 32, 2019. Citation on page 29.

ARAUJO, V.; BALABIN, H.; HURTADO, J.; SOTO, A.; MOENS, M.-F. How relevant is selective memory population in lifelong language learning? **arXiv preprint arXiv:2210.00940**, 2022. Citation on page 66.

BAE, H.; BROPHY, E.; CHAN, R. H.; CHEN, B.; FENG, F.; GRAFFIETI, G.; GOEL, V.; HAO, X.; HAN, H.; KANAGARAJAH, S. *et al.* Iros 2019 lifelong robotic vision: Object recognition challenge [competitions]. **IEEE Robotics & Automation Magazine**, IEEE, v. 27, n. 2, p. 11–16, 2020. Citation on page 83.

BANSAL, A.; SIKKA, K.; SHARMA, G.; CHELLAPPA, R.; DIVAKARAN, A. Zero-shot object detection. In: **Proceedings of the European Conference on Computer Vision (ECCV)**. [S.l.: s.n.], 2018. p. 384–400. Citation on page 63.

BAR, A.; WANG, X.; KANTOROV, V.; REED, C. J.; HERZIG, R.; CHECHIK, G.; ROHRBACH, A.; DARRELL, T.; GLOBERSON, A. Detreg: Unsupervised pretraining with region priors for object detection. **arXiv preprint arXiv:2106.04550**, 2021. Citations on pages 36 and 61.

BEAULIEU, S.; FRATI, L.; MICONI, T.; LEHMAN, J.; STANLEY, K. O.; CLUNE, J.; CHENEY, N. Learning to continually learn. **arXiv preprint arXiv:2002.09571**, 2020. Citations on pages 24, 35, and 36.

BELOUADAH, E.; POPESCU, A.; KANELLOS, I. A comprehensive study of class incremental learning algorithms for visual tasks. **Neural Networks**, Elsevier, v. 135, p. 38–54, 2021. Citation on page 34.

BENDALE, A.; BOULT, T. Towards open world recognition. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2015. p. 1893–1902. Citation on page 62.

BUZZEGA, P.; BOSCHINI, M.; PORRELLO, A.; CALDERARA, S. Rethinking experience replay: a bag of tricks for continual learning. In: IEEE. **2020 25th International Conference on Pattern Recognition (ICPR)**. [S.l.], 2021. p. 2180–2187. Citation on page 66.

CACCIA, L.; PINEAU, J. Special: Self-supervised pretraining for continual learning. **arXiv preprint arXiv:2106.09065**, 2021. Citations on pages 35 and 36.

CACCIA, M.; RODRIGUEZ, P.; OSTAPENKO, O.; NORMANDIN, F.; LIN, M.; PAGE-CACCIA, L.; LARADJI, I. H.; RISH, I.; LACOSTE, A.; VÁZQUEZ, D. *et al.* Online fast adaptation and knowledge accumulation (osaka): a new approach to continual learning. **Advances in Neural Information Processing Systems**, v. 33, p. 16532–16545, 2020. Citation on page 36.

CARION, N.; MASSA, F.; SYNNAEVE, G.; USUNIER, N.; KIRILLOV, A.; ZAGORUYKO, S. End-to-end object detection with transformers. In: SPRINGER. **European conference on computer vision**. [S.l.], 2020. p. 213–229. Citation on page 29.

CARON, M.; MISRA, I.; MAIRAL, J.; GOYAL, P.; BOJANOWSKI, P.; JOULIN, A. Unsupervised learning of visual features by contrasting cluster assignments. **Advances in Neural Information Processing Systems**, v. 33, p. 9912–9924, 2020. Citation on page 36.

CARON, M.; TOUVRON, H.; MISRA, I.; JÉGOU, H.; MAIRAL, J.; BOJANOWSKI, P.; JOULIN, A. Emerging properties in self-supervised vision transformers. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.: s.n.], 2021. p. 9650–9660. Citations on pages 11 and 30.

CERMELLI, F.; MANCINI, M.; BULO, S. R.; RICCI, E.; CAPUTO, B. Modeling the background for incremental learning in semantic segmentation. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2020. p. 9233–9242. Citation on page 62.

CHAUDHRY, A.; RANZATO, M.; ROHRBACH, M.; ELHOSEINY, M. Efficient lifelong learning with a-gem. **arXiv preprint arXiv:1812.00420**, 2018. Citation on page 74.

CHEN, J.; WANG, S.; CHEN, L.; CAI, H.; QIAN, Y. Incremental detection of remote sensing objects with feature pyramid and knowledge distillation. **IEEE Transactions on Geoscience and Remote Sensing**, IEEE, 2020. Citations on pages 49, 51, and 52.

CHEN, K.; WANG, J.; PANG, J.; CAO, Y.; XIONG, Y.; LI, X.; SUN, S.; FENG, W.; LIU, Z.; XU, J. *et al.* Mmdetection: Open mmlab detection toolbox and benchmark. **arXiv preprint arXiv:1906.07155**, 2019. Citation on page 77.

CHEN, L.; YU, C.; CHEN, L. A new knowledge distillation for incremental object detection. In: IEEE. **2019 International Joint Conference on Neural Networks (IJCNN)**. [S.l.], 2019. p. 1–7. Citations on pages 24, 49, 52, 53, 57, 58, 86, and 87.

CHEN, Z.; LIU, B. Lifelong machine learning. **Synthesis Lectures on Artificial Intelligence and Machine Learning**, Morgan & Claypool Publishers, v. 12, n. 3, p. 1–207, 2018. Citation on page 31.

CLUNE, J. Ai-gas: Ai-generating algorithms, an alternate paradigm for producing general artificial intelligence. **arXiv preprint arXiv:1905.10985**, 2019. Citation on page 30.

DELANGE, M.; ALJUNDI, R.; MASANA, M.; PARISOT, S.; JIA, X.; LEONARDIS, A.; SLABAUGH, G.; TUYTELAARS, T. A continual learning survey: Defying forgetting in classification tasks. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, 2021. Citations on pages 24, 32, 34, 35, 44, and 79.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. Imagenet: A large-scale hierarchical image database. In: IEEE. **2009 IEEE conference on computer vision and pattern recognition**. [S.l.], 2009. p. 248–255. Citation on page 37.

DESHPANDE, A. **A Beginner‘s Guide To Understanding Convolutional Neural Networks**. 2017. <https://adeshpande3.github.io/A-Beginner’s-Guide-To-Understanding-Convolutional-Neural-Networks/>. Accessed: 2019-11-25. Citations on pages 11 and 29.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. **arXiv preprint arXiv:1810.04805**, 2018. Citations on pages 27 and 31.

DÍAZ-RODRÍGUEZ, N.; LOMONACO, V.; FILLIAT, D.; MALTONI, D. Don’t forget, there is more than forgetting: new metrics for continual learning. **arXiv preprint arXiv:1810.13166**, 2018. Citations on pages 22, 23, and 33.

DONG, N.; ZHANG, Y.; DING, M.; LEE, G. H. Bridging non co-occurrence with unlabeled in-the-wild data for incremental object detection. **Advances in Neural Information Processing Systems**, v. 34, 2021. Citations on pages 52, 55, 57, and 59.

DOSOVITSKIY, A.; BEYER, L.; KOLESNIKOV, A.; WEISSENBORN, D.; ZHAI, X.; UNTERTHINER, T.; DEHGHANI, M.; MINDERER, M.; HEIGOLD, G.; GELLY, S. *et al.* An image is worth 16x16 words: Transformers for image recognition at scale. **arXiv preprint arXiv:2010.11929**, 2020. Citation on page 29.

DOUILLARD, A.; CHEN, Y.; DAPOGNY, A.; CORD, M. Plop: Learning without forgetting for continual semantic segmentation. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2021. p. 4040–4050. Citation on page 62.

DOUILLARD, A.; LESORT, T. Continuum: Simple management of complex continual learning scenarios. **arXiv preprint arXiv:2102.06253**, 2021. Citation on page 61.

DUAN, K.; BAI, S.; XIE, L.; QI, H.; HUANG, Q.; TIAN, Q. Centernet: Keypoint triplets for object detection. In: **Proceedings of the IEEE/CVF international conference on computer vision**. [S.l.: s.n.], 2019. p. 6569–6578. Citation on page 40.

EVERINGHAM, M.; GOOL, L. V.; WILLIAMS, C. K.; WINN, J.; ZISSERMAN, A. The pascal visual object classes (voc) challenge. **International journal of computer vision**, Springer, v. 88, n. 2, p. 303–338, 2010. Citation on page 40.

FALCON, W. **PyTorch Lightning**. 2019. Available: <https://github.com/Lightning-AI/lightning>. Citation on page 68.

FLENNERHAG, S.; RUSU, A. A.; PASCANU, R.; VISIN, F.; YIN, H.; HADSELL, R. Meta-learning with warped gradient descent. **arXiv preprint arXiv:1909.00025**, 2019. Citations on pages 36 and 55.

GALLARDO, J.; HAYES, T. L.; KANAN, C. Self-supervised training enhances online continual learning. **arXiv preprint arXiv:2103.14010**, 2021. Citations on pages 24 and 36.

GE, Z.; LIU, S.; WANG, F.; LI, Z.; SUN, J. Yolox: Exceeding yolo series in 2021. **arXiv preprint arXiv:2107.08430**, 2021. Citation on page 40.

GIRSHICK, R. Fast r-cnn. In: **Proceedings of the IEEE international conference on computer vision**. [S.l.: s.n.], 2015. p. 1440–1448. Citations on pages 11, 37, 38, and 51.

GIRSHICK, R.; DONAHUE, J.; DARRELL, T.; MALIK, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2014. p. 580–587. Citation on page 37.

GRAUMAN, K.; WESTBURY, A.; BYRNE, E.; CHAVIS, Z.; FURNARI, A.; GIRDHAR, R.; HAMBURGER, J.; JIANG, H.; LIU, M.; LIU, X. *et al.* Ego4d: Around the world in 3,000 hours of egocentric video. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2022. p. 18995–19012. Citations on pages 84 and 85.

GROSSBERG, S. T. **Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control**. [S.l.]: Springer Science & Business Media, 2012. Citation on page 21.

GUAN, L.; WU, Y.; ZHAO, J.; YE, C. Learn to detect objects incrementally. In: IEEE. **2018 IEEE Intelligent Vehicles Symposium (IV)**. [S.l.], 2018. p. 403–408. Citations on pages 52, 54, and 57.

GUPTA, A.; DOLLAR, P.; GIRSHICK, R. Lvis: A dataset for large vocabulary instance segmentation. In: **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. [S.l.: s.n.], 2019. p. 5356–5364. Citations on pages 40 and 97.

HADSELL, R.; RAO, D.; RUSU, A. A.; PASCANU, R. Embracing change: Continual learning in deep neural networks. **Trends in cognitive sciences**, Elsevier, 2020. Citations on pages 11, 22, 23, 44, and 73.

HAO, Y.; FU, Y.; JIANG, Y.-G. Take goods from shelves: A dataset for class-incremental object detection. In: **Proceedings of the 2019 on International Conference on Multimedia Retrieval**. [S.l.: s.n.], 2019. p. 271–278. Citations on pages 49, 52, 53, and 66.

HAO, Y.; FU, Y.; JIANG, Y.-G.; TIAN, Q. An end-to-end architecture for class-incremental object detection with knowledge distillation. In: IEEE. **2019 IEEE International Conference on Multimedia and Expo (ICME)**. [S.l.], 2019. p. 1–6. Citations on pages 24, 51, 52, 56, and 58.

HAQ, Q. M. ul; RUAN, S.-J.; HAQ, M. A.; KARAM, S.; SHIEH, J. L.; CHONDRO, P.; GAO, D.-Q. An incremental learning of yolov3 without catastrophic forgetting for smart city applications. **IEEE Consumer Electronics Magazine**, IEEE, 2021. Citations on pages 52, 53, and 73.

HASSABIS, D.; KUMARAN, D.; SUMMERFIELD, C.; BOTVINICK, M. Neuroscience-inspired artificial intelligence. **Neuron**, Elsevier, v. 95, n. 2, p. 245–258, 2017. Citations on pages 21 and 60.

HAYES, T. L.; KANAN, C. Selective replay enhances learning in online continual analogical reasoning. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2021. p. 3502–3512. Citations on pages 66 and 67.

HAYES, T. L.; KEMKER, R.; CAHILL, N. D.; KANAN, C. New metrics and experimental paradigms for continual learning. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops**. [S.l.: s.n.], 2018. p. 2031–2034. Citation on page 33.

HAYES, T. L.; KRISHNAN, G. P.; BAZHENOV, M.; SIEGELMANN, H. T.; SEJNOWSKI, T. J.; KANAN, C. Replay in deep learning: Current approaches and missing biological elements. **Neural Computation**, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 33, n. 11, p. 2908–2950, 2021. Citations on pages 34 and 66.

HAYKIN, S. **Neural networks and learning machines, 3/E**. [S.l.]: Pearson Education India, 2010. Citation on page 27.

HONDA, H. **Digging into detectron 2**. Medium, 2022. Available: <https://medium.com/@hirotoschwert/digging-into-detectron-2-47b2e794fabd>. Citations on pages 11 and 38.

HOSPEDALES, T.; ANTONIOU, A.; MICAELLI, P.; STORKEY, A. Meta-learning in neural networks: A survey. **arXiv preprint arXiv:2004.05439**, 2020. Citation on page 35.

HU, D.; YAN, S.; LU, Q.; LANQING, H.; HU, H.; ZHANG, Y.; LI, Z.; WANG, X.; FENG, J. How well does self-supervised pre-training perform with streaming data? In: **International Conference on Learning Representations**. [S.l.: s.n.], 2022. Citations on pages 36 and 61.

HUANG, G.; LARADJI, I.; VAZQUEZ, D.; LACOSTE-JULIEN, S.; RODRIGUEZ, P. A survey of self-supervised and few-shot object detection. **arXiv preprint arXiv:2110.14711**, 2021. Citations on pages 36, 37, 39, and 61.

HUNG, C.-Y.; TU, C.-H.; WU, C.-E.; CHEN, C.-H.; CHAN, Y.-M.; CHEN, C.-S. Compacting, picking and growing for unforgetting continual learning. **Advances in Neural Information Processing Systems**, v. 32, 2019. Citation on page 34.

JAVED, K.; WHITE, M. Meta-learning representations for continual learning. **Advances in Neural Information Processing Systems**, v. 32, 2019. Citation on page 36.

JING, L.; TIAN, Y. Self-supervised visual feature learning with deep neural networks: A survey. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 43, n. 11, p. 4037–4058, 2020. Citation on page 36.

JOSEPH, K.; KHAN, S.; KHAN, F. S.; BALASUBRAMANIAN, V. N. Towards open world object detection. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2021. p. 5830–5840. Citations on pages 32, 52, 54, 56, 57, 59, 61, 62, and 67.

KHAN, S.; NASEER, M.; HAYAT, M.; ZAMIR, S. W.; KHAN, F. S.; SHAH, M. Transformers in vision: A survey. **ACM Computing Surveys (CSUR)**, ACM New York, NY, 2021. Citation on page 29.

KIRKPATRICK, J.; PASCANU, R.; RABINOWITZ, N.; VENESS, J.; DESJARDINS, G.; RUSU, A. A.; MILAN, K.; QUAN, J.; RAMALHO, T.; GRABSKA-BARWINSKA, A. *et al.* Overcoming catastrophic forgetting in neural networks. **Proceedings of the national academy of sciences**, National Acad Sciences, v. 114, n. 13, p. 3521–3526, 2017. Citations on pages 34 and 74.

KJ, J.; RAJASEGARAN, J.; KHAN, S.; KHAN, F. S.; BALASUBRAMANIAN, V. N. Incremental object detection via meta-learning. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, 2021. Citations on pages 52, 55, 56, 57, 58, 59, and 61.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. **Advances in neural information processing systems**, v. 25, p. 1097–1105, 2012. Citations on pages 21 and 27.

KUNDU, J. N.; VENKATESH, R. M.; VENKAT, N.; REVANUR, A.; BABU, R. V. Class-incremental domain adaptation. In: SPRINGER. **European Conference on Computer Vision**. [S.l.], 2020. p. 53–69. Citation on page 44.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group, v. 521, n. 7553, p. 436, 2015. Citations on pages 21, 27, 29, and 40.

LECUN, Y.; BENGIO, Y. *et al.* Convolutional networks for images, speech, and time series. **The handbook of brain theory and neural networks**, v. 3361, n. 10, p. 1995, 1995. Citation on page 21.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, Ieee, v. 86, n. 11, p. 2278–2324, 1998. Citation on page 28.

LECUN, Y.; DENKER, J.; SOLLA, S. Optimal brain damage. **Advances in neural information processing systems**, v. 2, 1989. Citation on page 75.

LECUN, Y.; TOURESKY, D.; HINTON, G.; SEJNOWSKI, T. A theoretical framework for back-propagation. In: **Proceedings of the 1988 connectionist models summer school**. [S.l.: s.n.], 1988. v. 1, p. 21–28. Citation on page 21.

LI, D.; CAO, G.; XU, Y.; CHENG, Z.; NIU, Y. Technical report for iccv 2021 challenge sslad-track3b: Transformers are better continual learners. **arXiv preprint arXiv:2201.04924**, 2022. Citation on page 44.

LI, D.; TASCI, S.; GHOSH, S.; ZHU, J.; ZHANG, J.; HECK, L. Rilod: Near real-time incremental learning for object detection at the edge. In: **Proceedings of the 4th ACM/IEEE Symposium on Edge Computing**. [S.l.: s.n.], 2019. p. 113–126. Citations on pages 15, 52, 55, 57, 73, 74, and 77.

LI, H.; KADAV, A.; DURDANOVIC, I.; SAMET, H.; GRAF, H. P. Pruning filters for efficient convnets. **arXiv preprint arXiv:1608.08710**, 2016. Citation on page 75.

LI, K.; WAN, G.; CHENG, G.; MENG, L.; HAN, J. Object detection in optical remote sensing images: A survey and a new benchmark. **ISPRS Journal of Photogrammetry and Remote Sensing**, Elsevier, v. 159, p. 296–307, 2020. Citation on page 51.

LI, P.; LI, Y.; CUI, H.; WANG, D. Class-incremental few-shot object detection. **arXiv preprint arXiv:2105.07637**, 2021. Citation on page 62.

LI, W.; WU, Q.; XU, L.; SHANG, C. Incremental learning of single-stage detectors with mining memory neurons. In: IEEE. **2018 IEEE 4th International Conference on Computer and Communications (ICCC)**. [S.l.], 2018. p. 1981–1985. Citations on pages 15, 52, 54, 56, 57, 58, 74, 75, 77, 92, and 96.

LI, Z.; ARORA, S. An exponential learning rate schedule for deep learning. **arXiv preprint arXiv:1910.07454**, 2019. Citation on page 66.

LI, Z.; HOIEM, D. Learning without forgetting. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 40, n. 12, p. 2935–2947, 2017. Citations on pages 34, 51, and 74.

LIN, T.-Y.; DOLLÁR, P.; GIRSHICK, R.; HE, K.; HARIHARAN, B.; BELONGIE, S. Feature pyramid networks for object detection. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2017. p. 2117–2125. Citations on pages 37 and 38.

LIN, T.-Y.; GOYAL, P.; GIRSHICK, R.; HE, K.; DOLLÁR, P. Focal loss for dense object detection. In: **Proceedings of the IEEE international conference on computer vision**. [S.l.: s.n.], 2017. p. 2980–2988. Citations on pages 39 and 73.

LIN, T.-Y.; MAIRE, M.; BELONGIE, S.; HAYS, J.; PERONA, P.; RAMANAN, D.; DOLLÁR, P.; ZITNICK, C. L. Microsoft coco: Common objects in context. In: SPRINGER. **European conference on computer vision**. [S.l.], 2014. p. 740–755. Citations on pages 40, 41, and 85.

LIU, C.; WU, H. Channel pruning based on mean gradient for accelerating convolutional neural networks. **Signal Processing**, Elsevier, v. 156, p. 84–91, 2019. Citation on page 75.

LIU, L.; KUANG, Z.; CHEN, Y.; XUE, J.-H.; YANG, W.; ZHANG, W. Incdet: In defense of elastic weight consolidation for incremental object detection. **IEEE transactions on neural networks and learning systems**, IEEE, 2020. Citations on pages 49, 52, 54, 57, 58, and 59.

LIU, W.; ANGUELOV, D.; ERHAN, D.; SZEGEDY, C.; REED, S.; FU, C.-Y.; BERG, A. C. Ssd: Single shot multibox detector. In: SPRINGER. **European conference on computer vision**. [S.l.], 2016. p. 21–37. Citation on page 39.

LIU, X.; YANG, H.; RAVICHANDRAN, A.; BHOTIKA, R.; SOATTO, S. Multi-task incremental learning for object detection. **arXiv preprint arXiv:2002.05347**, 2020. Citations on pages 52 and 53.

LIU, Y.; SCHIELE, B.; VEDALDI, A.; RUPPRECHT, C. Continual detection transformer for incremental object detection. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2023. p. 23799–23808. Citations on pages 66 and 67.

LIU, Z.; LI, J.; SHEN, Z.; HUANG, G.; YAN, S.; ZHANG, C. Learning efficient convolutional networks through network slimming. In: **Proceedings of the IEEE international conference on computer vision**. [S.l.: s.n.], 2017. p. 2736–2744. Citation on page 54.

LIU, Z.; LIN, Y.; CAO, Y.; HU, H.; WEI, Y.; ZHANG, Z.; LIN, S.; GUO, B. Swin transformer: Hierarchical vision transformer using shifted windows. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.: s.n.], 2021. p. 10012–10022. Citation on page 94.

LOMONACO, V.; MALTONI, D. Core50: a new dataset and benchmark for continuous object recognition. In: PMLR. **Conference on Robot Learning**. [S.l.], 2017. p. 17–26. Citation on page 32.

LOMONACO, V.; PELLEGRINI, L.; COSSU, A.; CARTA, A.; GRAFFIETI, G.; HAYES, T. L.; LANGE, M. D.; MASANA, M.; POMPONI, J.; VEN, G. M. van de *et al.* Avalanche: an end-to-end library for continual learning. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2021. p. 3600–3610. Citations on pages 61 and 85.

LOMONACO, V.; PELLEGRINI, L.; RODRIGUEZ, P.; CACCIA, M.; SHE, Q.; CHEN, Y.; JODELET, Q.; WANG, R.; MAI, Z.; VAZQUEZ, D. *et al.* Cvpr 2020 continual learning in computer vision competition: Approaches, results, current challenges and future directions. **Artificial Intelligence**, Elsevier, v. 303, p. 103635, 2022. Citation on page 83.

LOPEZ-PAZ, D.; RANZATO, M. Gradient episodic memory for continual learning. In: **Advances in Neural Information Processing Systems**. [S.l.: s.n.], 2017. p. 6467–6476. Citations on pages 33 and 35.

LOWE, D. G. Distinctive image features from scale-invariant keypoints. **International journal of computer vision**, Springer, v. 60, n. 2, p. 91–110, 2004. Citation on page 37.

LUO, J.-H.; WU, J. An entropy-based pruning method for cnn compression. **arXiv preprint arXiv:1706.05791**, 2017. Citation on page 75.

MALLYA, A.; DAVIS, D.; LAZEBNIK, S. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In: **Proceedings of the European Conference on Computer Vision (ECCV)**. [S.l.: s.n.], 2018. p. 67–82. Citation on page 34.

MALLYA, A.; LAZEBNIK, S. Packnet: Adding multiple tasks to a single network by iterative pruning. In: **Proceedings of the IEEE conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2018. p. 7765–7773. Citation on page 74.

MENEZES, A. G.; MOURA, G. de; ALVES, C.; CARVALHO, A. C. de. Continual object detection: A review of definitions, strategies, and challenges. **Neural Networks**, Elsevier, 2023. Citation on page 67.

MICHIELI, U.; ZANUTTIGH, P. Incremental learning techniques for semantic segmentation. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops**. [S.l.: s.n.], 2019. p. 0–0. Citation on page 62.

MIRZADEH, S. I.; CHAUDHRY, A.; HU, H.; PASCANU, R.; GORUR, D.; FARAJTABAR, M. Wide neural networks forget less catastrophically. **arXiv preprint arXiv:2110.11526**, 2021. Citations on pages 24 and 73.

MIRZADEH, S. I.; FARAJTABAR, M.; PASCANU, R.; GHASEMZADEH, H. Understanding the role of training regimes in continual learning. **arXiv preprint arXiv:2006.06958**, 2020. Citations on pages 24 and 56.

MUNDT, M.; HONG, Y. W.; PLIUSHCH, I.; RAMESH, V. A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning. **arXiv preprint arXiv:2009.01797**, 2020. Citations on pages 31, 32, and 62.

MUNDT, M.; LANG, S.; DELFOSSE, Q.; KERSTING, K. Cleva-compass: A continual learning evaluation assessment compass to promote research transparency and comparability. **arXiv preprint arXiv:2110.03331**, 2021. Citation on page 32.

NIKOLIC, J.; BURRI, M.; REHDER, J.; LEUTENEGGER, S.; HUERZELER, C.; SIEGWART, R. A uav system for inspection of industrial facilities. In: IEEE. **2013 IEEE Aerospace Conference**. [S.l.], 2013. p. 1–8. Citation on page 89.

NORMANDIN, F.; GOLEMO, F.; OSTAPENKO, O.; RODRIGUEZ, P.; RIEMER, M. D.; HURTADO, J.; KHETARPAL, K.; ZHAO, D.; LINDEBORG, R.; LESORT, T. *et al.* Sequoia: A software framework to unify continual learning research. **arXiv preprint arXiv:2108.01005**, 2021. Citation on page 32.

OORD, A. v. d.; DIELEMAN, S.; ZEN, H.; SIMONYAN, K.; VINYALS, O.; GRAVES, A.; KALCHBRENNER, N.; SENIOR, A.; KAVUKCUOGLU, K. Wavenet: A generative model for raw audio. **arXiv preprint arXiv:1609.03499**, 2016. Citations on pages 27 and 29.

PADILLA, R.; NETTO, S. L.; SILVA, E. A. D. A survey on performance metrics for object-detection algorithms. In: IEEE. **2020 international conference on systems, signals and image processing (IWSSIP)**. [S.l.], 2020. p. 237–242. Citations on pages 11, 40, and 41.

PARISI, G. I.; KEMKER, R.; PART, J. L.; KANAN, C.; WERMTER, S. Continual lifelong learning with neural networks: A review. **Neural Networks**, Elsevier, v. 113, p. 54–71, 2019. Citations on pages 22 and 44.

PELLEGRINI, L.; GRAFFIETI, G.; LOMONACO, V.; MALTONI, D. Latent replay for real-time continual learning. **arXiv preprint arXiv:1912.01100**, 2019. Citation on page 35.

____. Latent replay for real-time continual learning. In: IEEE. **2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. [S.l.], 2020. p. 10203–10209. Citation on page 65.

PELLEGRINI, L.; ZHU, C.; XIAO, F.; YAN, Z.; CARTA, A.; LANGE, M. D.; LOMONACO, V.; SUMBALY, R.; RODRIGUEZ, P.; VAZQUEZ, D. 3rd continual learning workshop challenge on egocentric category and instance level object understanding. **arXiv preprint arXiv:2212.06833**, 2022. Citations on pages 65, 83, and 88.

PENG, C.; ZHAO, K.; LOVELL, B. C. Faster ilod: Incremental learning for object detectors based on faster rcnn. **Pattern Recognition Letters**, Elsevier, v. 140, p. 109–115, 2020. Citations on pages 24, 44, 47, 52, 53, 56, 57, 59, 61, 67, and 87.

PENG, C.; ZHAO, K.; MAKSOUD, S.; LI, M.; LOVELL, B. C. Sid: Incremental learning for anchor-free object detection via selective and inter-related distillation. **Computer Vision and Image Understanding**, Elsevier, p. 103229, 2021. Citations on pages 52, 53, 55, 57, 58, 59, and 60.

PEREZ-RUA, J.-M.; ZHU, X.; HOSPEDALES, T. M.; XIANG, T. Incremental few-shot object detection. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2020. p. 13846–13855. Citation on page 62.

PHAM, Q.; LIU, C.; HOI, S. Dualnet: Continual learning, fast and slow. **Advances in Neural Information Processing Systems**, v. 34, 2021. Citation on page 36.

PRABHU, A.; HAMMOUD, H. A. A. K.; DOKANIA, P. K.; TORR, P. H.; LIM, S.-N.; GHANEM, B.; BIBI, A. Computationally budgeted continual learning: What does matter? In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2023. p. 3698–3707. Citation on page 72.

QU, H.; RAHMANI, H.; XU, L.; WILLIAMS, B.; LIU, J. Recent advances of continual learning in computer vision: An overview. **arXiv preprint arXiv:2109.11369**, 2021. Citation on page 60.

RAJASEGARAN, J.; KHAN, S.; HAYAT, M.; KHAN, F. S.; SHAH, M. itaml: An incremental task-agnostic meta-learning approach. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2020. p. 13588–13597. Citation on page 36.

RAMAKRISHNAN, K.; PANDA, R.; FAN, Q.; HENNING, J.; OLIVA, A.; FERIS, R. Relationship matters: Relation guided knowledge transfer for incremental learning of object detectors. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops**. [S.l.: s.n.], 2020. p. 250–251. Citations on pages 49, 52, 53, 57, 58, and 59.

RATCLIFF, R. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. **Psychological review**, American Psychological Association, v. 97, n. 2, p. 285, 1990. Citations on pages 21 and 65.

REBUFFI, S.-A.; KOLESNIKOV, A.; SPERL, G.; LAMPERT, C. H. icarl: Incremental classifier and representation learning. In: **Proceedings of the IEEE conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2017. p. 2001–2010. Citations on pages 31, 35, and 66.

REDMON, J.; DIVVALA, S.; GIRSHICK, R.; FARHADI, A. You only look once: Unified, real-time object detection. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 779–788. Citation on page 39.

REN, S.; HE, K.; GIRSHICK, R.; SUN, J. Faster r-cnn: Towards real-time object detection with region proposal networks. **Advances in neural information processing systems**, v. 28, 2015. Citations on pages 31 and 37.

ROBINS, A. Catastrophic forgetting, rehearsal and pseudorehearsal. **Connection Science**, Taylor & Francis, v. 7, n. 2, p. 123–146, 1995. Citations on pages 22 and 34.

ROLNICK, D.; AHUJA, A.; SCHWARZ, J.; LILLICRAP, T.; WAYNE, G. Experience replay for continual learning. **Advances in Neural Information Processing Systems**, v. 32, 2019. Citation on page 65.

ROSS, D. A.; LIM, J.; LIN, R.-S.; YANG, M.-H. Incremental learning for robust visual tracking. **International journal of computer vision**, Springer, v. 77, n. 1, p. 125–141, 2008. Citation on page 22.

RUMELHART, D. E. Reducing interference in distributed memories through episodic gating. **From learning theory to connectionist theory**, Psychology Press, v. 1, p. 227, 1992. Citation on page 34.

RUSU, A. A.; RABINOWITZ, N. C.; DESJARDINS, G.; SOYER, H.; KIRKPATRICK, J.; KAVUKCUOGLU, K.; PASCANU, R.; HADSELL, R. Progressive neural networks. **arXiv preprint arXiv:1606.04671**, 2016. Citations on pages 34 and 54.

SCHOFIELD, O. B.; LORENZEN, K. H.; EBEID, E. Cloud to cable: A drone framework for autonomous power line inspection. In: **2020 23rd Euromicro Conference on Digital System Design (DSD)**. [S.l.: s.n.], 2020. p. 503–509. Citation on page 89.

SHAHEEN, K.; HANIF, M. A.; HASAN, O.; SHAFIQUE, M. Continual learning for real-world autonomous systems: Algorithms, challenges and frameworks. **arXiv preprint arXiv:2105.12374**, 2021. Citations on pages 22, 23, 35, and 44.

____. Continual learning for real-world autonomous systems: Algorithms, challenges and frameworks. **Journal of Intelligent & Robotic Systems**, Springer, v. 105, n. 1, p. 1–32, 2022. Citation on page 83.

SHIEH, J.-L.; HAQ, M. A.; KARAM, S.; CHONDRO, P.; GAO, D.-Q.; RUAN, S.-J. *et al.* Continual learning strategy in one-stage object detection framework based on experience replay for autonomous driving vehicle. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 20, n. 23, p. 6777, 2020. Citations on pages 35, 49, 52, 53, 57, and 66.

SHIN, H.; LEE, J. K.; KIM, J.; KIM, J. Continual learning with deep generative replay. **Advances in neural information processing systems**, v. 30, 2017. Citation on page 35.

SHMELKOV, K.; SCHMID, C.; ALAHARI, K. Incremental learning of object detectors without catastrophic forgetting. In: **Proceedings of the IEEE international conference on computer vision**. [S.l.: s.n.], 2017. p. 3400–3409. Citations on pages 15, 22, 44, 45, 46, 51, 52, 56, 57, 58, 59, 60, 68, 74, 75, 77, and 86.

SILVER, D. L. Machine lifelong learning: challenges and benefits for artificial general intelligence. In: SPRINGER. **International conference on artificial general intelligence**. [S.l.], 2011. p. 370–375. Citation on page 30.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. **arXiv preprint arXiv:1409.1556**, 2014. Citation on page 29.

SMITH, L. N. Cyclical learning rates for training neural networks. In: IEEE. **2017 IEEE winter conference on applications of computer vision (WACV)**. [S.l.], 2017. p. 464–472. Citation on page 88.

TAKAYA, K.; OHTA, H.; KROUMOV, V.; SHIBAYAMA, K.; NAKAMURA, M. Development of uav system for autonomous power line inspection. In: IEEE. **2019 23rd International Conference on System Theory, Control and Computing (ICSTCC)**. [S.l.], 2019. p. 762–767. Citation on page 90.

THRUN, S. **Lifelong Learning: A Case Study.** [S.l.], 1995. Citations on pages 30 and 31.

TIAN, Z.; SHEN, C.; CHEN, H.; HE, T. Fcos: A simple and strong anchor-free object detector. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, 2020. Citations on pages 40, 73, and 85.

TOUVRON, H.; CORD, M.; DOUZE, M.; MASSA, F.; SABLAYROLLES, A.; JÉGOU, H. Training data-efficient image transformers & distillation through attention. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2021. p. 10347–10357. Citation on page 29.

UIJLINGS, J. R.; SANDE, K. E. V. D.; GEVERS, T.; SMEULDERS, A. W. Selective search for object recognition. **International journal of computer vision**, Springer, v. 104, n. 2, p. 154–171, 2013. Citation on page 37.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. **Advances in neural information processing systems**, v. 30, 2017. Citation on page 29.

VEN, G. M. Van de; TOLIAS, A. S. Three scenarios for continual learning. **arXiv preprint arXiv:1904.07734**, 2019. Citation on page 32.

VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: IEEE. **Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001**. [S.l.], 2001. v. 1, p. I–I. Citation on page 37.

WALKER, S. J. Big data: A revolution that will transform how we live, work, and think. **International Journal of Advertising**, Routledge, v. 33, n. 1, p. 181–183, 2014. Citation on page 21.

WANG, J.; JIANG, T.; CUI, Z.; CAO, Z. Filter pruning with a feature map entropy importance criterion for convolution neural networks compressing. **Neurocomputing**, Elsevier, v. 461, p. 41–54, 2021. Citation on page 75.

WANG, J.; WANG, X.; SHANG-GUAN, Y.; GUPTA, A. Wanderlust: Online continual object detection in the real world. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.: s.n.], 2021. p. 10829–10838. Citations on pages 49 and 52.

WOHLIN, C. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: **Proceedings of the 18th international conference on evaluation and assessment in software engineering**. [S.l.: s.n.], 2014. p. 1–10. Citation on page 45.

WU, X.; SAHOO, D.; HOI, S. C. Recent advances in deep learning for object detection. **Neurocomputing**, Elsevier, v. 396, p. 39–64, 2020. Citation on page 37.

WU, Y.; CHEN, Y.; WANG, L.; YE, Y.; LIU, Z.; GUO, Y.; FU, Y. Large scale incremental learning. In: **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. [S.l.: s.n.], 2019. p. 374–382. Citation on page 66.

WU, Y.; KIRILLOV, A.; MASSA, F.; LO, W.-Y.; GIRSHICK, R. **Detectron2**. 2019. <https://github.com/facebookresearch/detectron2>. Citation on page 56.

XIA, G.-S.; BAI, X.; DING, J.; ZHU, Z.; BELONGIE, S.; LUO, J.; DATCU, M.; PELILLO, M.; ZHANG, L. Dota: A large-scale dataset for object detection in aerial images. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2018. p. 3974–3983. Citation on page 51.

YANG, D.; ZHOU, Y.; WANG, W. Multi-view correlation distillation for incremental object detection. **arXiv preprint arXiv:2107.01787**, 2021. Citations on pages 44, 50, 52, 53, 57, 58, and 59.

YANG, D.; ZHOU, Y.; WU, D.; MA, C.; YANG, F.; WANG, W. Two-level residual distillation based triple network for incremental object detection. **arXiv preprint arXiv:2007.13428**, 2020. Citations on pages 52, 54, 57, 58, and 59.

YANG, S.; SUN, P.; JIANG, Y.; XIA, X.; ZHANG, R.; YUAN, Z.; WANG, C.; LUO, P.; XU, M. Objects in semantic topology. **arXiv preprint arXiv:2110.02687**, 2021. Citations on pages 52, 53, 57, 59, and 63.

YOON, J.; YANG, E.; LEE, J.; HWANG, S. J. Lifelong learning with dynamically expandable networks. **arXiv preprint arXiv:1708.01547**, 2017. Citations on pages 34 and 59.

ZENKE, F.; POOLE, B.; GANGULI, S. Continual learning through synaptic intelligence. In: PMLR. **International conference on machine learning**. [S.l.], 2017. p. 3987–3995. Citation on page 74.

ZHAI, J.; LIU, X. **Technical Report for Domain Incremental Object Detection**. 2021. Available at: <https://sslad2021.github.io/>. Citation on page 44.

ZHANG, H.; WANG, Y.; DAYOUB, F.; SUNDERHAUF, N. Varifocalnet: An iou-aware dense object detector. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2021. p. 8514–8523. Citation on page 94.

ZHANG, J.; ZHANG, J.; GHOSH, S.; LI, D.; TASCI, S.; HECK, L.; ZHANG, H.; KUO, C.-C. J. Class-incremental learning via deep model consolidation. In: **Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision**. [S.l.: s.n.], 2020. p. 1131–1140. Citations on pages 52, 55, and 57.

ZHANG, N.; SUN, Z.; ZHANG, K.; XIAO, L. Incremental learning of object detection with output merging of compact expert detectors. In: IEEE. **2021 4th International Conference on Intelligent Autonomous Systems (ICoIAS)**. [S.l.], 2021. p. 1–7. Citations on pages 52 and 54.

ZHOU, W.; CHANG, S.; SOSA, N.; HAMANN, H.; COX, D. Lifelong object detection. **arXiv preprint arXiv:2009.01129**, 2020. Citations on pages 51, 52, 56, 57, 58, and 59.

ZOU, Z.; SHI, Z.; GUO, Y.; YE, J. Object detection in 20 years: A survey. **arXiv preprint arXiv:1905.05055**, 2019. Citations on pages 13, 40, and 73.

ICMC USP
SÃO CARLOS