End of Studies Internship

Field of Study: Computer Vision
Scholar Year: 2022-2023

Anomaly Detection through Vision-Language Models

Confidentiality Notice
Non-confidential report and publishable on Internet

Author:
Pietro TANURE ONNIS

Promotion:
2023

ENSTA IP Paris Tutor:
Antoine MANZARENA

Host Organism Tutor:
Quoc Cuong LE

Internship from 13/04/2023 to 13/10/2023

Name of the host organism: XXII Group
Address: 13-15 Rue Jean Jaurès
Puteaux
France

Confidentiality Notice

This present document is not confidential. It can be communicated outside in paper format or distributed in electronic format.

Abstract

Artificial Intelligence is a field of computer science interested in creating systems capable of simulating behaviors typical to human intelligence: learning, generalization, understanding. Computer vision is a field of AI interested in using these techniques to interpret and understand the visual world, where models are trained with examples of images to be able to learn to detect objects and people, classify and segment images, generate new images, etc. A problem arrives when there are not many examples available of a certain anomalous category for us to train our model, detecting these anomalies is an important problem with many real world applications, like detecting animals on roads for autonomous vehicles, surveillance and security, we call this more generally the domain of out-of-distribution detection. The advances of computer power have allowed for more complex models to be developed, and more recently the field of NLP (Natural Language Processing) is intersecting more and more with computer vision, allowing the computer to learn from both text and image and thus have a more robust, general and deep understanding of scenes, objects, meaning and relationship. During this study we focused on applying vision-language models to the problem of detecting rare objects on image scenes.

Résumé

L'intelligence artificielle est un domaine de l'informatique qui s'intéresse à la création de systèmes capables de simuler des comportements propres à l'intelligence humaine : apprentissage, généralisation, compréhension. La vision par ordinateur est un domaine de l'IA qui s'intéresse à l'utilisation de ces techniques pour interpréter et comprendre le monde visuel, les modèles sont entraînés avec des exemples d'images pour pouvoir apprendre à détecter des objets et des personnes, classer et segmenter des images, générer de nouvelles images, etc. Un problème survient lorsqu'il n'y a pas beaucoup d'exemples disponibles d'une certaine catégorie anormale pour que nous entraîner notre modèle, la détection de ces anomalies est un problème important avec de nombreuses applications du monde réel, comme la détection d'animaux sur les routes pour les véhicules autonomes, la surveillance et la sécurité, nous appelons cela plus généralement le domaine de la détection hors distribution. Les progrès de la puissance informatique ont permis de développer des modèles plus complexes, et plus récemment, le domaine du NLP (Natural Language Processing) se croise de plus en plus avec la vision par ordinateur, permettant à l'ordinateur d'apprendre à la fois du texte et de l'image et ainsi d'avoir un compréhension plus robuste, générale et profonde des scènes, des objets, de la signification et des relations. Au cours de cette étude, nous nous sommes concentrés sur l'application de modèles de langage de vision au problème de la détection d'objets rares sur des scènes d'images.
Mots-clés: Vision par ordinateur, détection hors distribution, modèles de vision-language.

Contents

List of Tables

List of Figures

Part I

Introduction

A high-performance machine learning model depends on the architecture of the model itself and on the data it is trained on. Every architecture brings in a certain inductive bias, a set of assumptions about the relationship between inputs and outputs that makes the algorithm learn one pattern instead of another pattern. If a model has a smaller inductive bias it is able to generalize to a larger range of contexts but it also needs more data to be trained on and to learn. One of the reasons the transformer architecture [11] is dominating so many domains of artificial intelligence is that it has less inductive bias compared to Convolutional Neural Network (CNN) or Recurrent Neural Network (RNN), which were conceived to be suited to image and sequential respectively.

While this may seem like a good thing it imposes the challenge of finding good, reliable data that can be used to train the model to produce accurate predictions. However, real world data is often messy and full of issues, annotation errors exist even in widely used image datasets [12] and can have a huge impact on performance ("Garbage in, garbage out") [13]. The costs of labeling, re-labeling and verification is sometimes boundless, and astronomical.

Problems with the dataset can be particularly harmful when working with incidents that do not have many occurrences in the data. These incidents can be rare objects, e.g. fire, fire spark, litters, etc. They can be rare events (someone running in a shopping mall) or objects abnormal for a given context (unattended luggage). Identifying these anomalies might be critical and they might be specially difficult to identify due to the lack of training data which allows the model to generalize and learn to identify them.

This type of problem is referred to as Out-Of-Distribution (OOD) Detection: given a known dataset, the goal is to determine if a new sample belongs to the same distribution or is in some way atypical. This is a large family of problems and so it is necessary to conduct an overview of the state-of-the-art on the subject and frame the problematic of this study among the existing ones on the bibliography. These problems live under what is called the *open-world* assumption. Traditionally Machine Learning (ML) tasks focus on *closed-world* settings, where it is assumed complete knowledge of the system and it is stated that test instances can only be from the distributions seen during training, this is in opposition to the *open-world* setting where instances can come from outside of the training distribution, this is illustrated in fig. I.1.

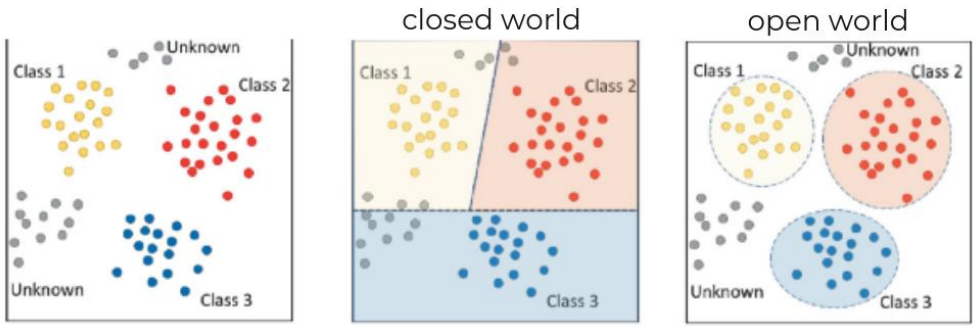


Figure I.1: *Closed-world* vs *Open-world* assumption

More recently there has been an increase in interest in multi-modal models that combine vision and language modalities. Being trained on image and text, these models have shown superior performance in challenging tasks such as visual question-answering, text-guided image generation and manipulation and image captioning [14]. They are often trained with millions or billions of image-text pairs and so show superior robustness and can be better transferred to real life applications including OOD detection [15].

One recent setting in the domain of VLM is that of Open Vocabulary Object Detection (OVD), where a detector is trained to identify and categorize not only a well known list of classes, but unseen objects defined by an unbounded vocabulary (i.e. image I.2). These models usually consist of taking an image encoder that was pre-trained in multimodal fashion with a text encoder and attach to it a detector with a modified head that contains an attention mechanism. In this study, we approach the task of Out-of-Distribution detection through the setting of open-vocabulary detection, we propose a novel YOLO-based model adapted for industrial applications (such as real-time deployment) using encoders pre-trained multimodally from text and image.

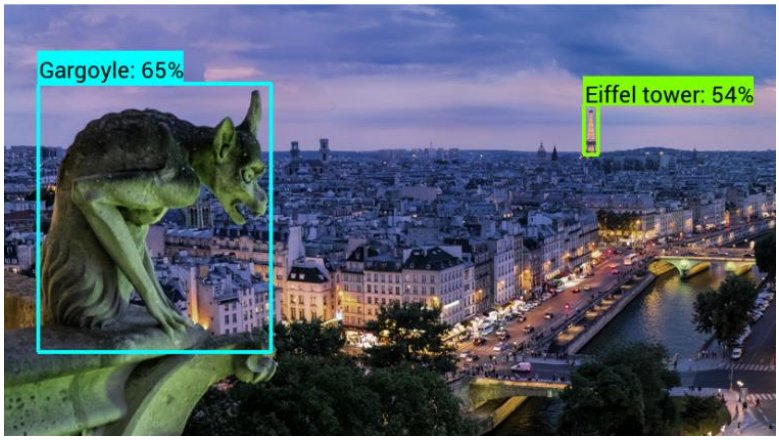


Figure I.2: Open Vocabulary Object Detection (OVD) inference example. Image from Arandjelović et al. [2]

Context of Internship

This study was done during my internship at XXII Group, which is a french scale-up computer vision software publisher company based in Puteaux, France. XXII built a platform capable of performing Real-Time video analysis from CCTV cameras, having many applications:

- Security purposes: detecting abnormal events, alerting in real-time
- Logistical analysis: object counting, waiting-time measurement

During my internship, I worked in the R&D department of the company responsible for doing research and developing new models, methods, and solutions to address the company's challenges. In specific the company observed that some rare objects like fire, unattended luggage and litter, are not detected with the same performance as more common classes like person, car, bike, etc. The possibility of detecting these unknown incidents could have multiple applications in deployment and data collection for further model training. The company defined the following prospectives for this internship:

1. Reduction of the cost in collecting and labeling data of rare classes
2. Real-time anomaly detection on the platform

Part II

Related Work

II.1 Generalized Out-of-Distribution Detection

Anomalies are events that somehow deviate from the normality, "normality" here is defined in a statistical sense as being instances or collections of data that rarely occur in the dataset and whose features differ significantly from most of the data, it is of interest to be able to identify these anomalies, whether they were seen before during training or not, and sometimes even classify them among new classes.

There are many practical interests in computer vision in being able to identify OOD instances, for example in medical imaging it can used be to identify tumors, in autonomous driving it can be used to identify strange objects or animals on the road (Fig. II.1), in security and surveillance camera systems it can identify potential threats in public or private spaces, etc.

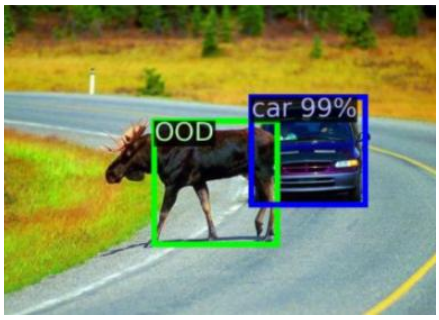


Figure II.1: Out-of-Distribution Detection inference example

This constitutes a whole family of different problems which include Anomaly Detection (AD), Novelty Detection (ND), One-Class Classification (OCC), Out-Of-Distribution (OOD) Detection, Open-Set Recognition (OSR), Novel Category Discovery (NCD), Outlier Detection [3]. The model is trained with a set of known classes in a labeled dataset $\mathcal{D}^l$ and has to assign the labels to possibly seen and unseen classes in a unlabeled dataset $\mathcal{D}^u$ during inference. More generally the problem can be formalized as: let $\mathcal{X}$ be the input (sensory) space and $\mathcal{Y}$ be the label (semantic) space, given a dataset $\mathcal{D}$ which comprises a labeled part $\mathcal{D}^l = \{(x_i^l, y_i^l)\}_{i=1}^N \in \mathcal{X} \times \mathcal{Y}^l$ with a $\mathcal{C}^l$ number of classes and an unlabeled part $\mathcal{D}^u = \{(x_i^u, y_i^u)\}_{i=1}^M \in \mathcal{X} \times \mathcal{Y}^u$ with a $\mathcal{C}^u$ number of classes, such that $\mathcal{Y}^l \neq \mathcal{Y}^u$, the model does not have access to the labels in $\mathcal{D}^u$ and needs to assign a label $Y^u \in \mathcal{Y}^u$ to $X^u \in \mathcal{X}$. Each problem can be seen as a subset of this more general problem. We summarize the notations on the table II.1.

Notations	Meaning
$\mathcal{X}$	the feature space in R
X^l/X^u	the data samples of the labeled/unlabeled sets.
$P(X)$	the marginal distribution of X
$\mathcal{Y}^l/\mathcal{Y}^u$	the target spaces in $\mathbb{R}^{C^l}/\mathbb{R}^{C^u}$
C^l/C^u	the number of classes in the labeled/unlabeled sets.
Y^l/Y^u	the corresponding class labels of X^l/X^u
$\mathcal{D}^l/\mathcal{D}^u$	the labeled/unlabeled data domains, composed of a set of samples X and their corresponding class labels Y
N/M	the number of samples in $\mathcal{D}^l/\mathcal{D}^u$

Table II.1: Notations used in generalized Out-of-Distribution detection and their meanings (we keep the notation of Troisemaine et al. [1])

The literature often uses different terms interchangeably, for clarity the conventions defined in Yang et al. [3] are used, who did a comprehensive survey for Generalized OOD Detection and proposed a unified framework to view closely related tasks on the literature based on 4 taxonomies seen on figure II.2.

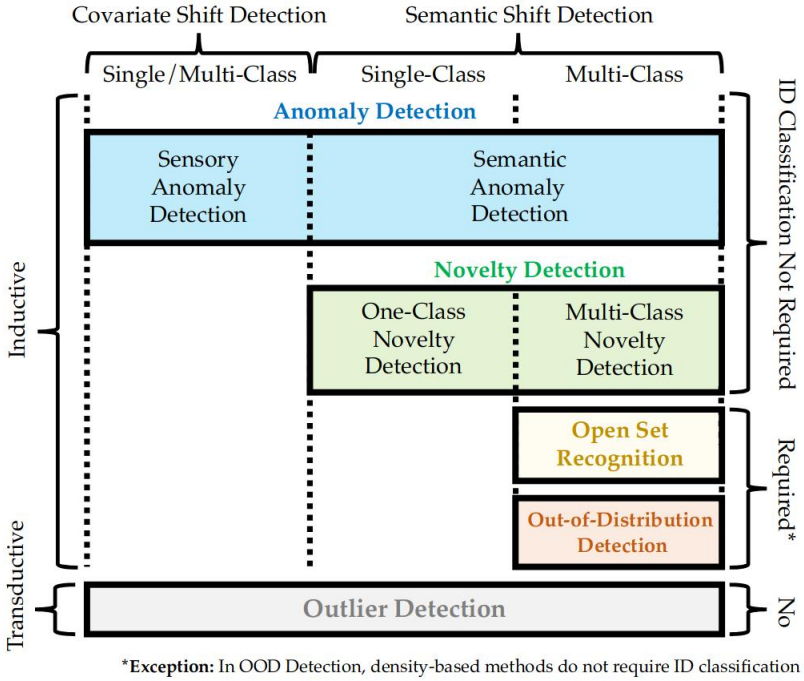


Figure II.2: Taxonomy of generalized OOD detection framework, illustrated by classification tasks. Image by: Yang et al. [3]

1. **Distribution shift:** the task focuses on detecting covariate shift ($P(X^l) \neq P(X^u)$) or semantic shift ($P(Y^l) \neq P(Y^u)$)
2. **In-Distribution classes:** the In-Distribution (ID) data contains one single class ($\mathcal{C}^l = 1$) or multiple classes ($\mathcal{C}^l > 1$).
3. **ID Classification:** Whether the task requires classifying the ID data
4. **Transductive or Inductive:** Transductive task requires all observations at once; inductive tasks follow the train-test scheme with the $\mathcal{D}^l$ and $\mathcal{D}^u$ respectively.

II.1.1 Anomaly and Novelty Detection

An "anomaly" is something that exhibits a distribution shift. It can be either from a covariate shift ($P(X^l) \neq P(X^u)$) or a semantic shift ($P(Y^l) \neq P(Y^u)$). A picture of a cat among pictures of dogs is a semantic shift and a drawing of a dog among pictures of real dogs is a covariate shift. Covariate shifts are commonly used to evaluate the generalization and robustness of a model, i.e. adversarial examples, domain shifts and style changes. Because a shift in $P(Y)$ also triggers a shift in $P(X)$ in practice most methods for identifying sensory and semantic shift are mutually inclusive, with some approaches being specialized in one of the two sub-tasks [3].

Different from other frameworks like Open-Set Recognition and Out-Of-Distribution Detection, the setting of Anomaly Detection (AD) treats In-Distribution (ID) samples as one thing even if they belong to many different classes ("person", "dog", "cat"...), it is not interested in doing classification of the ID samples, only assigning them either the label of "ID" or " OOD". Main applications of sensory AD are industrial inspection, image forensis, adversarial defense, forgery recognition of artworks. And for semantic AD it can be used for filtering data and for video surveillance.

Since anomalies are usually rare and examples are available in small quantities or not at all available, approaches for this setting are usually unsupervised or semi-supervised. Common approaches for AD models are: density-based where OOD test samples are rejected if they deviate from the main distribution [16]; reconstruction-based, where an encoder-decoder architecture is trained to accurately reconstruct the ID samples [17], this way an image of an OOD instance will have a bad reconstruction and can be identified; one-class classification OCC, mainly through the construction of a decision boundary between the ID and OOD samples [18].

Novelty Detection is similar to AD but is only interested in semantic shift, the main difference is a motivation one, ND sees novel classes not as erroneous or fraudulent, but as possible learning resources for a model that cannot possibly know all the classes it is shown. ND is also supposed to be fully unsupervised, while AD can have some abnormal training samples. Main applications of ND include incremental learning and dataset augmentation.

II.1.2 Open-Set Recognition and Out-Of-Distribution Detection

Open-Set Recognition (OSR) is interested in making classifiers robust and general enough to deal with real-world problems like incomplete information, limited data resources and imbalanced distribution. The task is to both: 1) identify samples from the trained classes (so called "known known classes"); 2) reject samples from never seen classes ("unknown unknown classes"). An example of OSR problem can be a face identification system where the model has to both identify each sample as being from a class (a specific individual) but also correctly classify any person it has not seen before as unknown.

OSR is closely related with other domains such as such Zero/few-shot learning, classification with reject option and Open-World Learning [19]. In Zero-shot classification the model is trained to predict classes with labeled positive training examples seen during training (known known classes) and also classes that only have side information (unknown known classes). An example would be training an image classifier with images of farm animals but also feeding the model texts about wild animals. This way by combining both its visual knowledge of a farm horse and a text that says "a zebra is like a horse with black and white stripes" the model is able to identify a zebra during test although

never having seen an image of one. Few-shot would be the same but the model instead has a few positively labeled image examples of the "unknown known classes" (like the zebra) on top of its images of "known known classes". Open-World learning is an evolution of OSR where the model is tasked with doing everything an OSR classifier does and also perform incremental learning to update the model continually as unknown unknown classes appear, without forgetting the known known classes.

Unlike AD and ND, OSR has the additional objective of identifying ID classes and so the majority of its models use a classification-based approach [20]; and some use a distance-based approach, where metric learning and contrastive learning are used to construct a latent space where samples from the same classes are clustered together while remaining separate from other classes [21][17].

Applications of OSR are in deploying real-world image classifiers in general, which can accurately identify the trained classes while also identifying OOD classes as they almost always exist in the real world.

Out-Of-Distribution (OOD) is very similar to OSR, difference is that OSR is interested in identifying semantic shift coming from the same dataset, while OOD methods normally consider ID as being classes in a given dataset and OOD as being samples drawn from a totally different dataset with non-overlapping classes. For example, training a model to identify ImageNet classes while rejecting samples from the MNIST dataset.

Like OSR, the majority of approaches are classification-based [22] [23] [24]. The key philosophical difference between OOD and OSR is that OOD is trained to identify samples from which the model does not want to or cannot generalize [3]. In this sense OOD covers a broader scope of tasks and its applications are usually in safety-critical situations, such as autonomous driving.

II.1.3 Outlier Detection

Outlier Detection is different from the previously discussed methods because there is no train/test split, the dataset is processed all at once, making the approach transductive. Models are usually: density-based, interested in modeling the probability distribution from the the data samples; distance-based performing metric-learning like techniques to cluster the samples from a specific class together [25]. OD is a vast domain, and its applications range from data mining, data pre-processing, video surveillance and network safety. It's important to note that the term outlier is a lot of times used interchangeably with anomaly and novelty so it's important to keep in mind the main frameworks and the task discussed.

II.1.4 Generalized Category Discovery

Generalized Category Discovery (GCD) is also a transductive approach where the model receives during training both a labeled dataset of known classes $\mathcal{D}^l = \{(x_i^l, y_i^l)\}_{i=1}^N \in \mathcal{X} \times \mathcal{Y}^l$ with a $\mathcal{C}^l$ number of classes and an unlabeled dataset $\mathcal{D}^u = \{(x_i^u)\}_{i=1}^M \in \mathcal{X}$ with $\mathcal{C}^u$ number of classes and is asked to label the instances in $\mathcal{D}^u$. Differently from the previously discussed problems, here the novel classes are not necessarily rare, and the dataset can be even overwhelmed by all these different classes. This setting was formalized by Vaze et al. [26] although the same setting had been explored before by the name of Open-World Semi-Supervised Learning [27]. GCD has the same setting of Novel Category Discovery but the later imposes a restriction of no intersection between $\mathcal{D}^u$ and $\mathcal{D}^l$. This setting

relative to the previously discussed settings is represented in fig II.3.

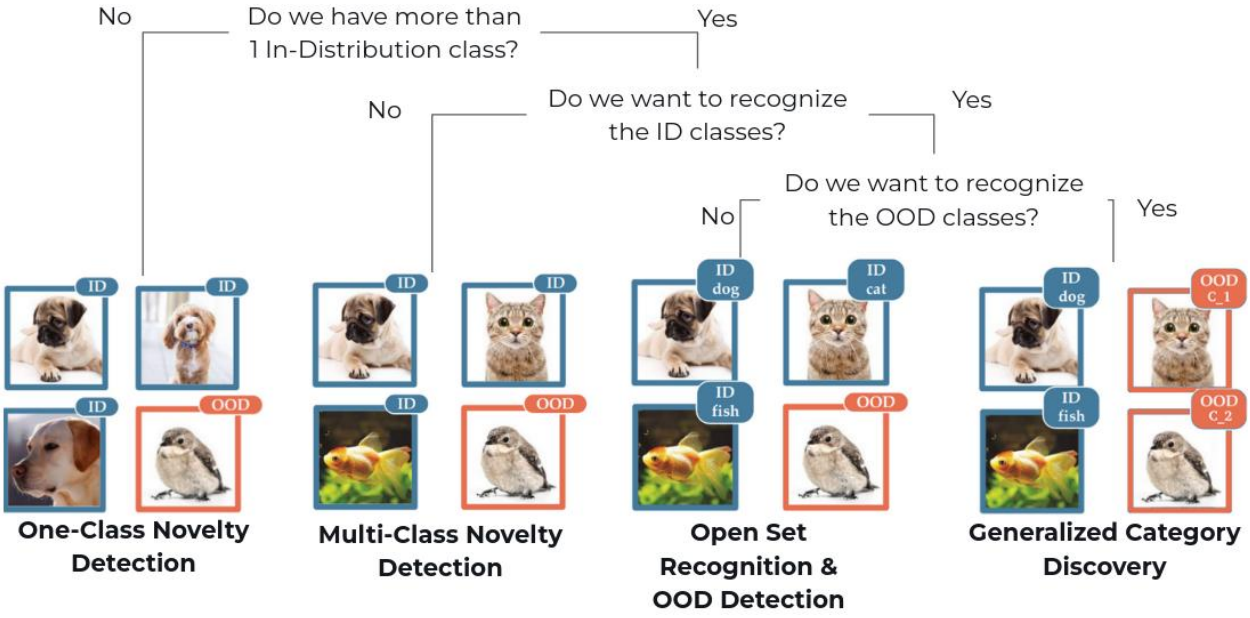


Figure II.3: Simplified diagram of main OOD detection frameworks

This setting has gained a lot a notoriety as well with recent models using distance-based approaches with contrastive learning [26], prototype learning [28] and prompt learning [29]. The domain is still a new one and so to this point the GCD and NCD models are only tested on relatively simple object oriented datasets, like CIFAR10, CIFAR100 and ImageNet, the models were not yet applied on scene-oriented datasets like MSCOCO, LVIS, etc. For this reason the domain is still not at the point of real world deployment where there are many classes on the same image (a lot of these classes not even annotated on the dataset), many different scene contexts and etc.

II.1.5 Summary

The domain of Generalized Out-Of-Distribution (OOD) Detection for computer vision is vast, as seen here there are a lot of frameworks, terminologies, and approaches, each one with its drawbacks and advantages. To recall the main ideas that were discussed, we provide a summary of the domains of OOD Detection adressed at the table II.2.

Need to ...	AD	ND	OSR	OOD	OD	NCD	GCD
recognize OOD instances	✓	✓	✓	✓	✓	✗	✓
have OOD samples during training	✓/✗	✗	✗	✓	✓	✓	✓
accurately classify known samples	✗	✗	✓	✓	✗	✗	✓
discover the new classes	✗	✗	✗	✗	✗	✓	✓

Table II.2: Summary of the domains in Generalized OOD Detection, including: Anomaly Detection (AD), Novelty Detection (ND), Open-Set Recognition (OSR), Out-Of-Distribution (OOD), Outlier Detection (OD), Novel Category Discovery (NCD), Generalized Category Discovery (GCD). Table from Troisemaine et al. [1]

AD and ND methods have been studied for a long time and so their models can be used on a variety of context, datasets and tasks, but they are not able to classify ID

classes. Since the platform developed by XXII Group is one capable of real-time object recognition, it works with a wide range of classes, scenes and backgrounds, so it is of interest to be able to classify existing ID classes. We turn our interest towards OOD and OSR models, which have good performance on scene oriented dataset like MSCOCO, they can classify ID classes but not distinguish between possible OOD classes, a feature that could be very desirable for annotation and recognition. The more complex tasks of NCD and GCD allow to both identify existing objects and classify new ones, but the domain is still quite new, having less than 2 years in the case of NCD and less than 1 year in the case of GCD, so the existing models have been tested for more object oriented datasets like CIFAR10, CIFAR100 and ImageNet, not yet being applied to complex scene oriented datasets like MSCOCO and LVIS (closer to the datasets used by XXII).

For these reasons, but mostly because of sheer curiosity about such a rapidly developing field we turned our interest towards Vision-Language Models (VLM), these models are trained on billions and millions of images, they can offer robustness and generalization, being able to solve a wide range of tasks on different domains.

II.2 Vision-Language Models

Visual recognition tasks like image classification, object recognition and semantic segmentation have posed a persistent challenge on the field of computer vision. The advent of deep learning brought great advances in the field, the focus was on developing Deep Neural Network (DNN) with architectures that imposed the appropriate inductive bias to solve the task. CNNs for example were inspired by the visual cortex in the human brain, where neurons respond to specific regions or receptive fields in the visual space, effectively capturing spatial relationships and patterns in images. The deep learning paradigm allowed models to be trained in supervised manner to many different tasks, but it comes with two drawbacks: the slow convergence of the DNN under the classical setup of "trained from scratch" and the collection of crowd-labeled and task-specific data to train these models which is very time-consuming [30].

Recently a new paradigm of *"pre-training and fine-tuning"* has shown to be very effective. Pre-training allows to construct a latent space dense enough to do transfer learning to many tasks. A model can be pre-trained on classification tasks on a wide enough dataset like ImageNet, it can also be pre-trained in self-supervised manner using data augmentation and contrastive learning to obtain a metric-like embedding space, or in the case of VLM models, in multimodal fashion harnessing text and image. The pre-trained model can be subsequently fine-tuned to many possible downstream tasks with great computational efficiency as many models don't even require fine-tuning to have good performance, being deployed directly in *zero-shot* manner.

In this context, Vision-language Models (VLM) are models that learn from both image and text, constructing a solid understanding of language, syntax, semantics, recognition and scene disposition which allows them to be capable of solving a wide range of tasks (fig II.4). These tasks can be multimodal, involving image and text, or unimodal (text-only / image-only).

II.2.1 Timeline

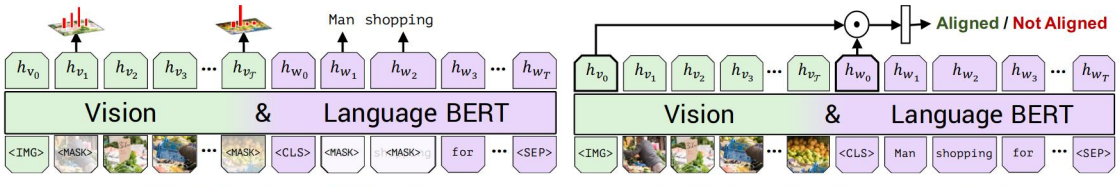
This domain has taken a lot of inspiration from the domain of Natural Language Processing to define its training objectives and architecture. After the original work by

Figure II.4: Task examples for VLM models. From left to write: Visual Question Answering (VQA), Visual Captioning (VC), Semantic Segmentation (SS), Grounding Referring Expressions (GRE)

Vaswani et al. [11] (2017) that introduced the transformer, studies began to try to apply it on images as well to create VLMs. We can't possibly offer a complete view of this field, but we offer a brief timeline of the main models for image-to-text tasks.

Li et al. [31] (2019) proposed **VilBERT**, it uses the encoder only transformer architecture BERT [32] to teach a model region-to-phrase grounding: that is the task of understanding the relationship between different parts of the images and words in the text. The transformer would receive a sequence of word embeddings alongside embeddings of regions of interest from images that were encoded by a CNN. Using this sequence the model is trained on a scene oriented dataset with two objectives: 1) Masked language modeling, in which some elements of text input are masked and must be predicted, while the vectors corresponding to image regions are not masked; 2) Image-text matching, where the model is provided a text segment consisting of two captions, one of them always corresponds to the image, while the other one has a 50% chance of corresponding to the image and 50% of being a randomly drawn caption, the model is then trained to distinguish these two situations.

VisualBERT (Lu et al. [4], 2019), had a similar approach of training a BERT architecture with text embeddings and embedding of regions of interest encoded by a CNN. It is trained on the Conceptual Captions dataset with 3.3M image-caption pairs, using two objectives (fig. II.5): 1) Masked multi-modal learning, in which elements of both the text and image input are masked and the model must predict the missing words and the semantic classes of the masked image patches; 2) Image-text matching, where the model is given an image-text pair and must decide whether they are aligned. In terms of architecture differently from VisualBERT it has a dual encoder architecture with separate transformers for text and image. It also introduces a novel co-attention mechanism, where the key and values computed by each of these two transformers is passed as input to the other transformer's multi-headed attention block, this way the model learn conjointly from image and text information. The resulting model is capable of many vision-and-language tasks: VQA, VCR, GRE, CBIR.



(a) Masked multi-modal learning (b) Multi-modal alignment prediction

Figure II.5: VilBERT training tasks visualized. Image by Lu et al. [4]

Dosovitskiy et al. [5] (2020) decided to apply the transformer directly on the image instead on pre-calculated CNN embeddings. They proposed the Vision Transformer (**ViT**), where the image is divided into patches and each patch is assigned a position encoding corresponding to its position on the image and are then linearly embedded, the resulting sequence is then fed through the standard Transformer encoder, as represented in figure II.6). The ViT showed that a single architecture, the transformer, could be used to treat both text and images directly. This opened the door for the development of a wide range of VLM with a wide range of learning objectives, architectures and capabilities over the following years.

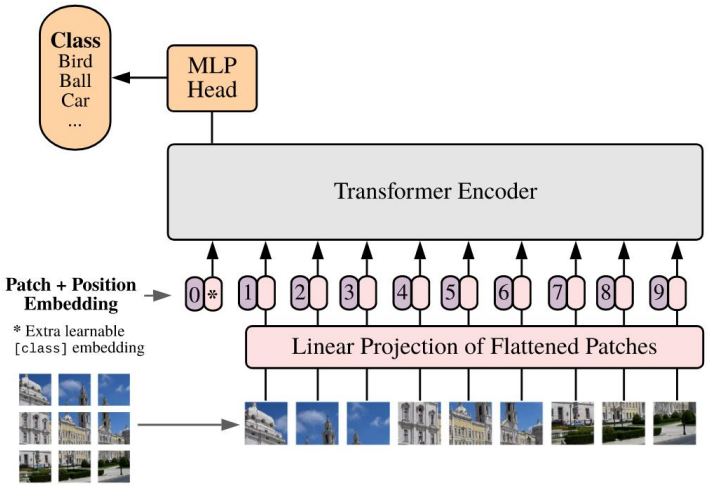


Figure II.6: Vision Transformer. Image by Dosovitskiy et al. [5]

Arguably the most influential article for this study is the work by Radford et al. [6](2021) which introduces the **CLIP** model, which trains a vision and a text encoder contrastively on image-caption pairs in order to construct a shared embedding space of the two encoders. CLIP's architecture is based on the dual encoder contrastive architecture proposed by Zhang et al. [33]. The affinity between an image and a text is given by the dot-product between the embeddings of the image and the text. During training the model is given a batch of image-caption pairs and asked to match each image with its correct caption. During inference The model is given a set of N text prompts of the type "A photo of a object.", each containing different category names, these prompts are then passed through the pre-trained text encoder to generate N text embeddings. These text embeddings are projected on the image embedding, and the class of the text prompt with the highest score is chosen as the category (image II.7). Training CLIP requires an immense amount of data, 400 million image-text pair were gathered from public available sources on the internet, the resulting dataset was never made available after. A transformer architecture was used for the text encoder and both CNN and transformers were trained as image encoder, with varying sizes: ResNet-50, ResNet-101 [34], three EfficientNet-style models and three Vision Transformers (ViT-B/32, ViT-B/16, ViT-L/14).

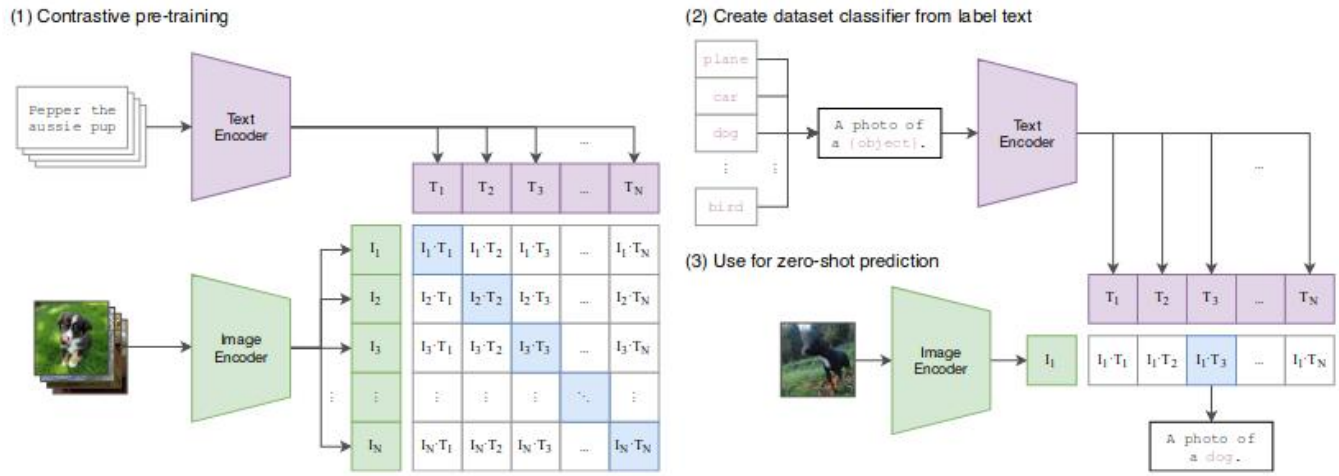


Figure II.7: Summary of CLIP training and inference. Image by Radford et al. [6]

Jia et al. [35](2021) also published a paper at the same time which used the same architecture and training, calling it the **ALIGN** model, they used an even bigger dataset of 1.8 billion images gathered from the internet following filtering procedures by Conceptual Captions [36].

Simple Visual Language Model (**SimVLM**) by Wang et al. [37](2021) is based on the basic Prefix Language Modeling objective from NLP training and has only one training objective. A transformer encoder receives both the image patches and the first half of a sentence and is asked to generate the second half of the sentence. The model adopts this simple architecture and training objective but is trained on the dataset used by ALIGN (1.8B image-text pairs) + the C4(text only) dataset. This massive amount of data allows the model to learn and generalize.

Differently from the previous models that go for a relatively simple training and a lot of data, Singh et al. [14] proposed **FLAVA** which uses 5 different training objectives and 3 encoders but manages to obtain comparable results with only 70M image and text pairs openly available online (compared to CLIP's 400M, ALIGN's 1.8B, SimVLM's 1.8B+). The model performs both unimodal tasks (text-only /image-only) and multimodal tasks, being more universal than previous approaches. To do this it uses an image encoder, a text encoder and a multimodal encoder, it is trained with 5 objectives: Masked language modeling (MLM), Masked Image Modeling (MIM), Masked Multimodal Modeling (MMM), Image-Text Matching (ITM), Global Contrastive (GC).

II.2.2 Summary

We make a summary of the different aspects of a VLM model in order to better synthesize the previous section:

1. **Feature extraction**: "CNN vs ViT vs Object-Detection"-based (whether you encode the whole image using a CNN or a ViT to obtain a feature vector, or use a detector to extract a RoI from the image containing a single object) / For the text feature extraction there are many transformer based architectures: BERT, RoBERTa, ALBERT, XLNet.

2. **Model architecture**: 1-stream vs 2-stream (whether the text and visual features enter the same or different encoders) / Encoder-only vs encoder-decoder (the cross-

modal representations from the encoder are directly fed into an output layer or they go through a decoder).

3. **Pre-training objectives**: Contrastive; Masked Language Modelling (MLM); Masked multi-modal modelling (MMM); Image-Text Matching (ITM); Prefix Language Modeling (PLM), etc.

4. **Pre-training datasets**: There are many available datasets like Conceptual Captions (CC3M), CC12M, Flickr30k, Visual Genome (VG), Red Caps, YFCC100M. Some models also used their own constructed dataset like CLIP and ALIGN.

5. **Downstream tasks**: Visual Question Answering (VQA); Visual Commonsense Reasoning (VCR); Object Recognition (OR); Grounding Referring Expressions (GRE); Category Recognition (CR); Vision-Language Retrieval (VLR); Visual Captioning (VC); Semantic Segmentation (SS), etc.

These aspects for the previously discussed methods can be seen in table II.3.

Model	**Image encoder**	**Text encoder**	**Decoder**	**Multimodal fusion**	**PT Objectives**	**PT datasets**	**Data size**
VisualBERT	BERT	BERT	no	single stream	MLM, ITM	COCO	113K
ViLBERT	BERT	BERT	no	dual stream	MMM, ITM	CC	3.3M
CLIP	ViT-L/14	Transformer	no	dual stream	VLC	CLIP*	400M
ALIGN	EfficientNet	BERT	no	dual stream	VLC	ALIGN*	1.8B
SimVLM	ResNet + Transformer	Transformer	yes	single stream	PrefixLM	ALIGN* + C4	1.8B+
FLAVA	ViT-B/16 + UniT	ViT-B/16 + UniT	yes	both	MLM, MIM, MMM, ITM, VLC	COCO, SBU, LN, CP, VG, WIT, CC12M, RC, YFCC100M	70M

Table II.3: Summary of vision-language models discussed.

II.3 Open-Vocabulary Object Detection

A new setting that is becoming very common with the development of large-scale contrastively pre-trained image-text models (ex. CLIP, ALIGN) is Open Vocabulary Object Detection (OVD) [38]. It is not only interested in detecting objects from a fixed number of classes, but also in being able to classify novel classes into a vast unrestricted semantic space by constructing a unified vision-language model.

The majority of OVD models are based on CLIP, they use a dual encoder architecture that was pre-trained using contrastive learning to align the feature spaces of the text and image encoders, allowing to make a direct comparison between image and text feature vectors by a simple scalar product. To allow for detection, the image encoder is used as the backbone of a detector, and in the last layer of the classification head, instead of k-way classification the head is modified so that the classification features can be projected directly upon pre-calculated text embeddings of the classes (fig: II.8).

While zero-shot detection methods learn a limited set of base classes and struggle to generalize to target classes, OVD models acquire a much larger vocabulary by learning from low-cost image-caption pairs. We'll provide a brief timeline of this field and go more in-depth on specific models that have influenced this paper.

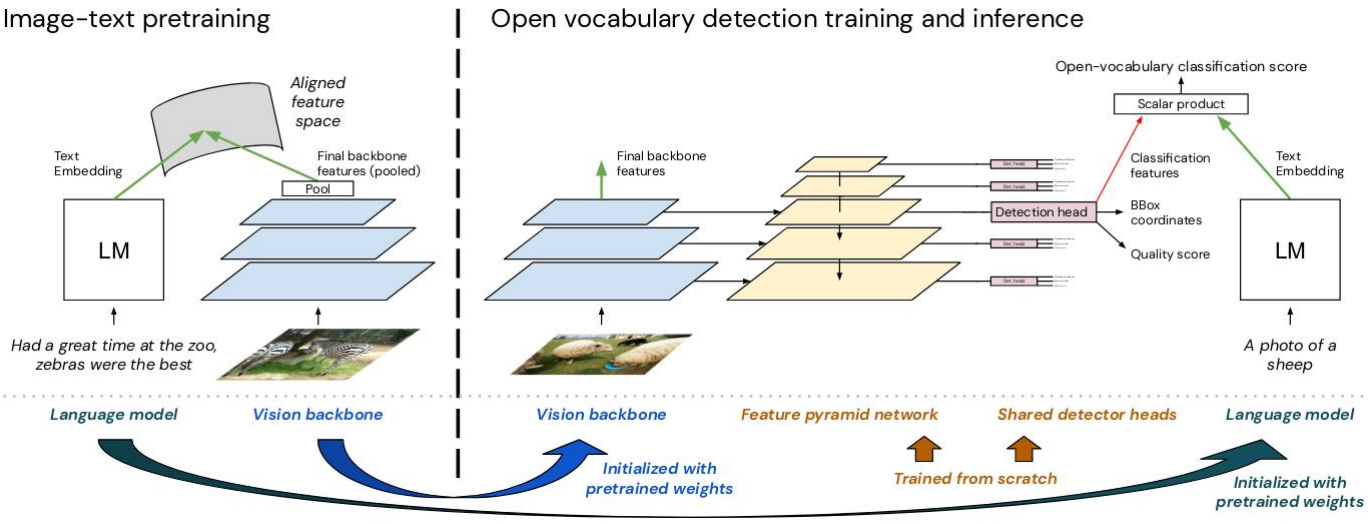


Figure II.8: Standard approach of Open Vocabulary Object Detection (OVD) pre-training and detection. Image by Arandjelović et al. [2]

II.3.1 Timeline

ViLD (Gu et al. [39] 2021) proposes to distill the image representation of a pre-trained teacher Vision-Language Model (VLM) like CLIP, into the detector. They train a student detector, whose embeddings of the detected regions are aligned with the text embeddings and also the image embeddings inferred by the teacher model. **DetPro** (Du et al. [23] 2022) improves upon ViLD by applying the idea of prompt optimization. Since VLM are very sensitive to the text or image prompt used [6] (i.e. using the text prompt "Photo of a object." and "object" give different performances, and slightly different RoI can make the model focus on different aspects of the scene) DetPro trains the detector to optimize the region proposal for the best classification score.

RegionCLIP (Zhong et al. [7] 2021) develops a region-text pre-training that leverages pre-trained VLMs on image-caption data and also train the classification head using distillation from a teacher VLM.

GLIP (Li et al. [40] 2021) uses a transformer architecture and reformulates object detection as a phrase grounding task, pre-training the model to correctly match many regions extracted from a single image with the text embedding corresponding to that region's class. It uses a variety of detection, grounding, and caption datasets for zero/few-shot object detection and differently from the other approaches it does deep cross-modality fusion between the text and image embeddings instead of only fusing text and image on the last layer like CLIP.

GLIPv2 (Zhang et al. [41] 2022) then uses inter-image contrastive by extracting regions from multiple images and introduces batch negative examples making the learning task harder. Since it maintains batch size it's more data efficient than GLIP.

OWL-ViT (Minderer et al. [42] 2022) finetunes an open-vocabulary detector with an encoder pre-trained in contrastive manner in various detection/grounding datasets. To adapt the image encoder for detection, they remove the token pooling and final projection layer of the ViT while box coordinates are obtained by passing the token representations through a small fully connected multi-layer neural network.

F-VLM (Kuo et al. [8] 2022) uses a frozen pre-trained VLM and trains only the

detector's head using an object detection dataset. It changes the detector's head during inference to allow for classes not seen during training. Using a frozen LM has the downside that the vision model is "forced" into the language-model "mould". OWL-ViT, GLIP and GLIPv2 also train the language model (LM) with smaller learning rate to prevent catastrophic forgetting.

UniDetector (Wang et al. [9] 2023) proposes to uses multiple dataset with heterogenous label spaces to train the detector. To promote the generalization to novel categories it does probability calibration and proposes a decoupled training that trains separetely the class-agnostic Region Proposal Network (RPN) and the class-aware classification head.

Three-ways (Arandjelović et al. [2] 2023) proposes three methods to improve feature alignment for OVD, being the current best performing algorithm. The first method is to augment the text embeddings by performing dropout to calculate multiple variations of the text embedding. Secondly the detector is modified to include gated shortcuts which garantee vision-text feature alignment during the beginning of training. Finally a self-training approach is used to leverage a larger corpus of image-text pairs for training.

II.3.2 RegionCLIP

Zhong et al. [7](2021) showed that while obtaining impressive performances on image classification CLIP was not as good in classifying image regions for object detection. That is because CLIP was trained to match an image as a whole to a text description without relating the image regions to words on the text through grounding. RegionCLIP was then proposed, it distill knowledge from CLIP to train a detector performing what was called "CLIP-guided region-text alignment". The first step is to take the pre-trained CLIP weights for the teacher image and text encoders, the second step is to choose an image-caption dataset (like CC3M) and extract regions of interest containing objects from these images using a detector. Key words present in the image captions are used to contruct many different prompts using the text encoder, where each class has its own text prompt. Then a new student image encoder is pre-trained using a contrastive loss to correctly assign each region to its respective text prompt and also with a distillation loss to match the cosine similarity between region-text obtained by the original CLIP teacher encoder. This way RegionCLIP learns to assign each image region to a specific class and also learns the dense semantic space of CLIP. Finally the third step is to take this new pre-trained image encoder, use it as the backbone of a detector and train it on object detection datasets like COCO or LVIS. This training can be seen in image II.9.

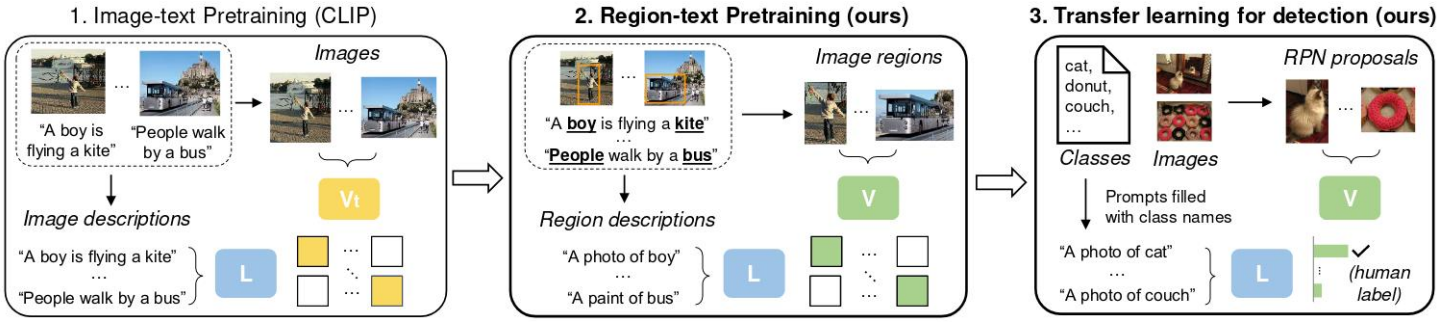


Figure II.9: RegionCLIP encoder pre-training and detector training. Vt is the teacher image encoder by CLIP and V is the new student encoder of RegionCLIP. Image by Zhong et al. [7]

RegionCLIP obtains better scores on image classification than CLIP.

II.3.3 F-VLM

Considering the high cost of pre-training, Kuo et al. [8](2022) explores the possibility of using a frozen pre-trained VLM to construct an open-vocabulary detector. Using a pre-trained CLIP encoder as the backbone of a Faster-RCNN detector [43] they only train the detector's head. The change comes in the detection scores $\mathbf{z}$, normally they are calculated by reducing the last fully connected layer to m neurons ($m-1$ classes and 1 background class) and performing a softmax operation. This approach doesn't support Open-vocabulary settings which require new categories to be added in test time. So instead the last fully connected layer is replaced with text embeddings $\mathbf{t}_j$ of base categories , and a cosine similarity operation is done between the region embeddings $\mathbf{r}_i$ obtained and the text embeddings, in a sense the detector's head is being taught to obtain an embedding space aligned with the text embedding space. The new detection scores are then given by equation II.1.

$$\mathbf{z}_i(\mathbf{r}_i) = Softmax(\frac{1}{\tau}[cos(\mathbf{r}_i, \mathbf{t}_1), ..., cos(\mathbf{r}_i, \mathbf{t}_m)]) \tag{II.1}$$

During inference the architecture is different because now new categories come at play, equation II.1 is applied to the region embeddings with both the old and new text embeddings to obtain $\mathbf{z}_i$. The model also takes the region bboxes proposed by the detector's head and performs a RoI Align operation which extracts the region corresponding to the bbox from the last layer of the feature map encoded by the backbone, obtaining a fixed size feature representation. This region's feature map is then passed through the attention pooling layer from the original pre-trained VLM to obtain a region embedding $\mathbf{v}_i$, this embedding is then projected upon both text embeddings $\mathbf{t}_j$ of the old m categories and of the new n categories using cosine similarity and taking the softmax like in equation II.1, obtaining $\mathbf{w}_i$. Finally the final score $\mathbf{s}_i$ for each region b is calculated as a geometric mean of $\mathbf{z}_i$ and $\mathbf{w}_i$. This architecture is illustrated on image II.10.

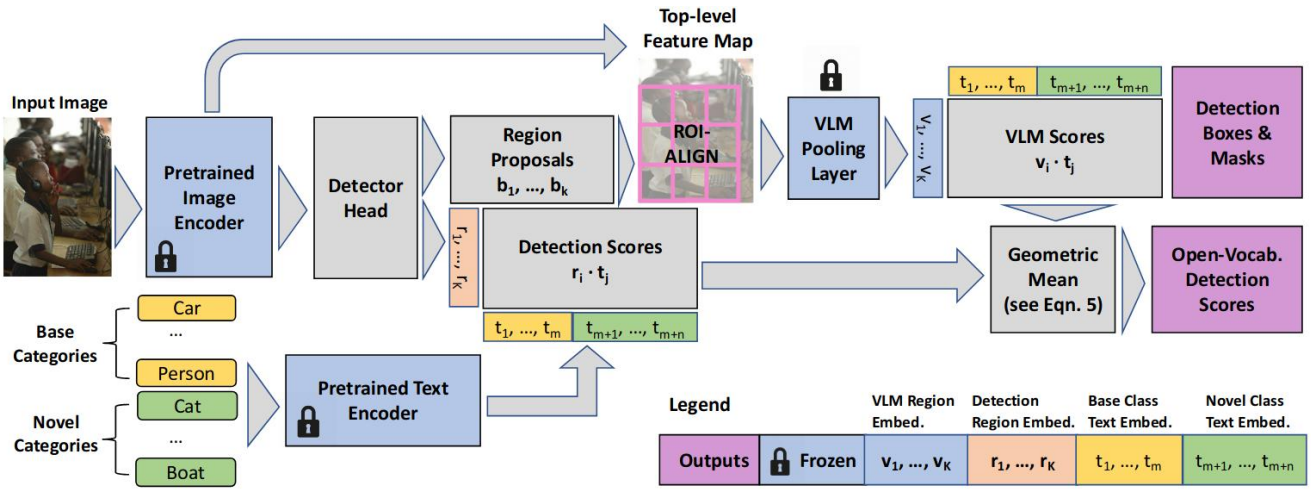


Figure II.10: F-VLM inference architecture. Image by Kuo et al. [8]

With this approach F-VLM attains competitive results with a fraction of the computation cost, since no pre-training was necessary, it also conserves the original visual-language

features and so outperforms RegionCLIP, ViLD [23] and Detic [44] in recognizing the rare categories in LVIS dataset.

II.3.4 UniDetector

Aiming at making a universal detector, Wang et al. [9] based their model on three critical points: 1) it must leverage images of multiple sources and heterogeneous label spaces for training; 2) generalizes to the open-world and keeps a balance between seen and unseen categories by using language and vision modalities; 3) keep generalization to novel categories through decoupled training and probability calibration.

The initial weights for the backbone to be fine-tuned are taken from the pre-trained backbone by RegionCLIP that was trained for region-text alignment. To allow generalization to both seen and unseen classes, a detector that proposes regions in a class-agnostic way is chosen, in this case FasterRCNN (image II.11). Since the RPN is class agnostic but RoI Head is not (having worse performance for novel classes), they are trained separetely in order for the head not to bias the performance of the RPN.

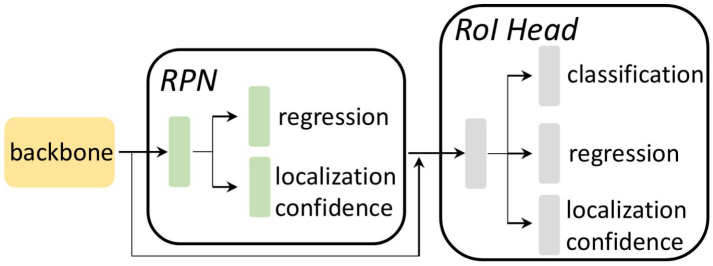


Figure II.11: Illustration of class-agnostic localization network used by UniDetector. Image by Wang et al. [9]

For the first point the model proposes different possibilities of model architectures to deal with the multiple datasets and label spaces, they find that sharing the same backbone but having a different trained head for each dataset give the best results since it promotes better feature extraction. Finally since only In-Distribution classes are seen during training the detector is biased towards these categories and gives higher confidence scores, so probability calibration is proposed for post-processing the predictions and balancing ID and OOD classes.

II.3.5 Summary

The field of Open Vocabulary Object Detection (OVD) is growing rapidly, and new approaches are being studied everyday, each with its drawbacks and advantages. One can pre-train a model on a large image-caption dataset and use it as the backbone of a detector [8][9][23][39][42] taking advantage of the dense feature obtained during the pre-training (which benefits OOD detection), or to directly pre-train it on the grounding task [7][40]. It is possible to apply either transformers-based [40][42] or CNN-based [8][9][23][39] architectures. The backbone can also be fine-tuned for object detection [9][42] which increases performance for ID classes but can distort the features for OOD detection. Also training the detector in two stage can be a good alternative to decouple the problem into a localization and a classification one, approaching it from two fronts.

Part III

Methodology

We now return to the real life context of XXII Group, the company observed that some rare objects are not detected with the same performance as more common classes. In order to train the model to identify these classes it is necessary to collect and annotate instances of these objects, but these kind of tasks are very costly. An approach that allows one not only to identify when something is OOD, but also classify it, might be very desirable for annotating new data.

Through the usage of a pre-trained model we can take advantage of models trained on much more data than we are capable of doing. Models that were pre-trained multimodally with image and text have shown superior robutness [15] and can narrow the gap between human and machine by allowing one to search specifically for something using a text prompt, helping to collect relevant data. For this reasons we turn ourselves towards the field of Open Vocabulary Object Detection that harnesses pre-trained VLM to do object detection.

Since we want to favor anomaly detection we keep the pre-trained encoder frozen like F-VLM, because fine-tuning it distorts the features learned during pre-training and so makes the model underperform OOD [45]. Also considering the fact that XXII does real time object recognition, an OVD that also prioritizes speed could also expand the possibilities of applications to not only data mining but also to deployment. For this reasons we choose a CNN encoder and not a transformer one, as well as a YOLO detector - in our vocabulary, an object detector is an object detection model - which is still faster than most other detectors while having good performance.

We summarize our reasoning and choices:

1. Pre-trained models can offer superior performance and generalization by being pre-trained on large amounts of data

2. Open Vocabulary Object Detection (OVD) harnesses multimodally pre-trained models for object detection, allowing text to image object detection.

3. To emphasize anomaly detection we keep the pre-trained model frozen

4. To emphasize inference speed and real time detection we adopt a CNN encoder and a YOLO model

The current models on the literature all use a two-stage detector for OVD, since they prioritize performance and not speed. But YOLO is a single-stage detector and so we need to adapt the standard approach illustrated in image II.8 for our study. UniDetector

showed that a decoupled training for a two-stage detector can be beneficial in order not to make its RoI head (which is not class agnostic) bias the RPN (which is class agnostic) during training. Our approach is similar, we first train a YOLO detector to do class agnostic object localization and later couple the trained model with a multi-modal head that allows to classify the localized objects from a text prompt.

III.1 Class agnostic Object Detection

The first step is to substitute the backbone of a detector with a CNN encoder pre-trained in contrastive fashion, and then to train this detector to do class agnostic object detection while keeping the new backbone frozen.

The original **YOLO** [46] is a single-stage detector, the image passes only once through the network and at the end a regression model proposes bounding boxes and a classification model predicts a class corresponding to each bbox. It does this by dividing the image into a grid, if the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes, a confidence scores that an object is in that bbox called *objectness*, and C conditional class probabilities. Finally this predictions are post-processed, the best bounding boxes are chosen and a class label is assigned to each. We observe this framework at image III.1.

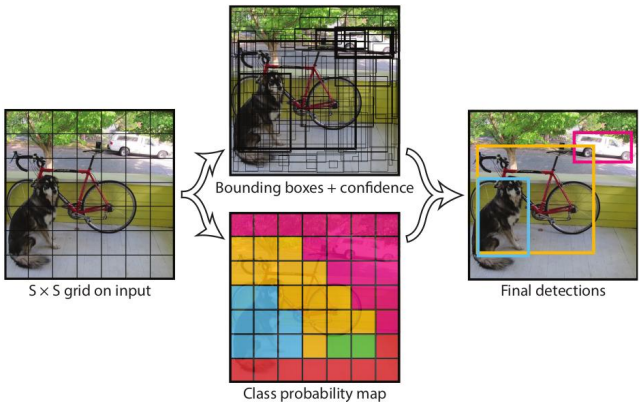


Figure III.1: YOLO model

The following sections go in detail into the implementation and training details. We used the framework *MMDetection*[10] for all implementations and trainings done in this study.

III.1.1 Implementation

Any single stage model could have been used for our implementation, we adopted YOLOv8 which is the most recent version of YOLO, being the fastest and most performing [47] yet. **YOLOv8** is based on the YOLOv5 model and introduces a series of changes, first it is an anchor-free method, this means it predicts directly the center of an object instead of the offset from a known anchor box. It also only predicts on bbox for each grid cell and gets rid of the *objectness* score, predicting directly the bbox among the possible classes. It uses both a Feature Pyramid Network (FPN) and a Path Aggregation Network (PAN). The FPN gradually reduces the spatial resolution of the input image while increasing the number of feature channels, creating a feature map capable of detecting objects at

different scales and resolutions. The PAN aggregates features from different levels of the network through skip connection capturing features at multiples scales and resolutions, increasing accuracy to detect different sizes and shapes [48]. This architecture is shown on image III.1.

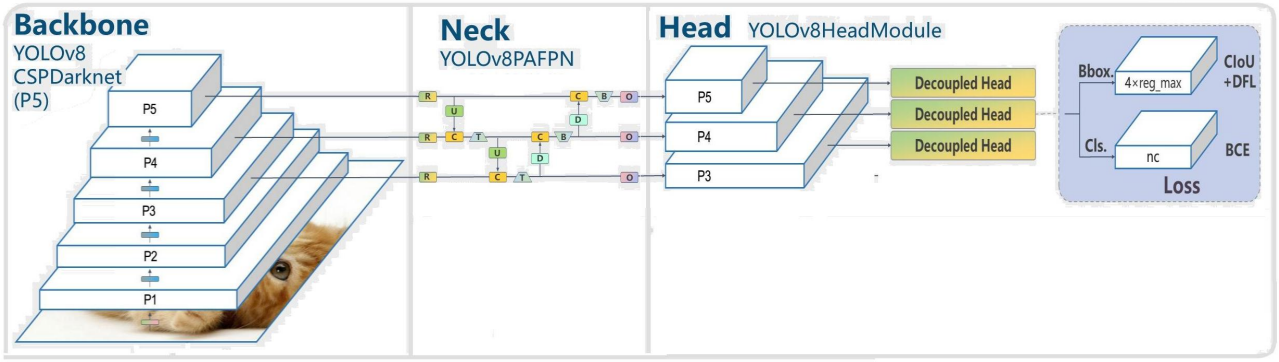


Figure III.2: Original YOLOv8 architecture. Image by MMDetection Contributors [10]

For the pre-trained CNN backbone backbone we adopt a ResNet50 pre-trained by CLIP. RN50 has 4 layers instead of 5 like the original's CSPDarknet, to keep the compatibility between the number of channels of each feature map expected by the neck and channels output by the RN50 backbone we use the large version of YOLOv8. The last layer of RN50 is not connected to the YOLO architecture that takes the feature maps of only the first three layers, but it is a part of our OVD model and will be used to generate the image embeddings to be compared directly to text, as it is explained in section III.2. Our new architecture with the RN50 backbone can be seen in image III.2, all other parts of the YOLOv8 model remain unchanged.

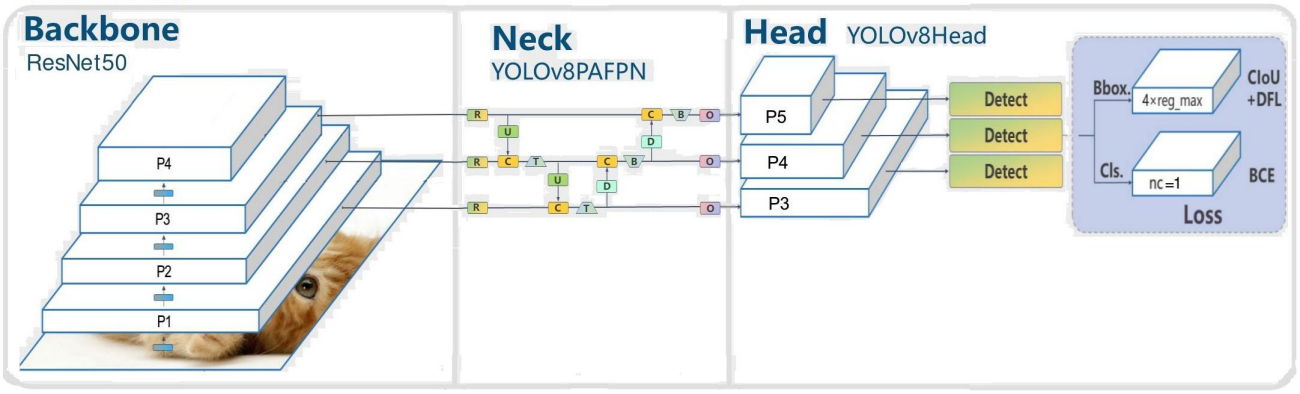


Figure III.3: YOLOv8 with pre-trained RN50 backbone.

III.1.2 Training

The training scheme remains the standard training for YOLOv8, including all its "bag of tricks" (i.e. image augmentation, transformations, optimization scheme, etc), it is an object detection training in batches.

To adapt YOLOv8 to be a class-agnostic detector we did a pre-processing phase on the labels of our dataset, we unified the class of all labels, so that all annotations belonged

to the same class, and so the number of classes on YOLOv8's classification head is $n_c = 1$. The pipeline of training is the following:

1. **Loading:**

   - **Load images** of training dataset

   - **Load annotations** of training dataset

2. **Preprocessing:**

   - **Unify the class labels** of the annotations

   - **Resize image** to 640x640 pixels the expected value expected by YOLOv8 large

   - **Normalize image** considering the values of the mean and standard deviation of the backbone, in this case the pre-trained encoder of the VLM.

3. **Augmentations and transformations:**

   - **Perform mosaic augmentations:** This augmentation combine four random crops of images (fig. III.4), combines classes that may not be seen together in your training set and changing the number of objects in your images. Mosaic augmentation was disables during the last 10 epochs of training.

   - **Perform random affine transformations:** These are geometric transformations that preserves lines and parallelism, it includes rotation, resizing and translation (fig. III.4).

   - **Perform mixup[49] augmentation:** Generates a weighted combination of random image pairs from the training data (fig. III.4), regularizing the neural network to favor simple linear behavior in-between training examples.

   - **Perform various operations** with 1% probability: MedianBlur, blur, gray-scale, flip, CLAHE.

Figure III.4: Pre-processing image transformations used in training

4. **Train the detector:**

   - **Forward propagation**: Pass a batch of images through the network and obtain predictions of bounding boxes and predicted class

- **Calculate loss:** Calculate the losses for the iteration, YOLOv8 uses three losses: the classification branch uses Binary Cross Entropy (BCE) loss and the regression branch uses Distribution Focal Loss (DFL) and Complete Intersection over Union Loss (CIoU). The generalized loss function is defined in equation III.1.

$$\mathcal{L}(\theta) = \frac{\lambda_{CIoU}}{N_{pos}}\mathcal{L}_{CIoU}(\theta) + \frac{\lambda_{BCE}}{N_{pos}}\mathcal{L}_{BCE}(\theta) + \frac{\lambda_{DFL}}{N_{pos}}\mathcal{L}_{DFL}(\theta) + \phi\|\theta\|_2^2 \tag{III.1}$$

Where N_{pos} is the total number of cells containing an object, $\lambda_{CIoU} = 7.5$, $\lambda_{BCE} = 0.5$ and $\lambda_{DFL} = 0.375$ are loss weights and θ are the model weights.

- **Backpropagation:** Propagate error backward through the network actualizing the weights θ. The optimizer used is Stochastic Gradient Descent (SGD), momentum $\beta = 0.937$, weight decay $\phi = 0.0005$ and learning rate $\eta = 0.01$. The procedure is given by eq III.2.

$$\begin{aligned} V^t &= \beta V^{t-1} + \nabla_\theta \mathcal{L}(\theta^{t-1}) \\ \theta^t &= \theta^{t-1} - \eta V^t \end{aligned} \tag{III.2}$$

The model was trained for 600 epochs on MSCOCO dataset.

III.2 YOLO-CLIP

Once we have a class agnostic detector that uses the VLM pre-trained encoder we can modify it to make it appropriate for Open Vocabulary Object Detection (OVD). The architecture that we have conceived is to use the bounding boxes proposed by YOLO and extract the corresponding regions of interests from the last layer of the backbone feature map. Then we do an attention pooling operation on this feature map (using an attention head that contains the weights of the pre-trained VLM) to reduce it to an embedding vector. This vector represents the image in the latent space, and because of the multimodal pre-training it is aligned with text embeddings from the text encoder.

To do the Region-of-Interest (RoI) extraction we used **RoI Align** [43], originally proposed for the Faster-RCNN model. It allows to extract from any region of the feature map an arbitrarily shaped tensor without loosing information due to quantization, this is particularly useful for Attention Pooling operations that required a fixed shaped tensor as input. It does this by performing bilinear interpolation on the feature map to obtain the cell value of the new tensor. This process is visualized in image III.5.

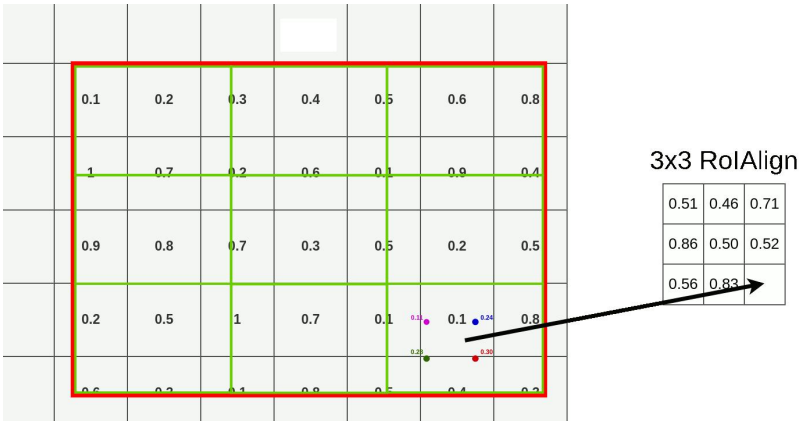


Figure III.5: RoI Align operation.

The attention pooling is a single layer of "transformer-style" multi-head QKV attention with the query conditioned on the global average-pooled representation of the image, it is initialized with the weights obtained during multi-modal pre-training. The complete architecture of our model, called YOLO-CLIP is seen on image III.6.

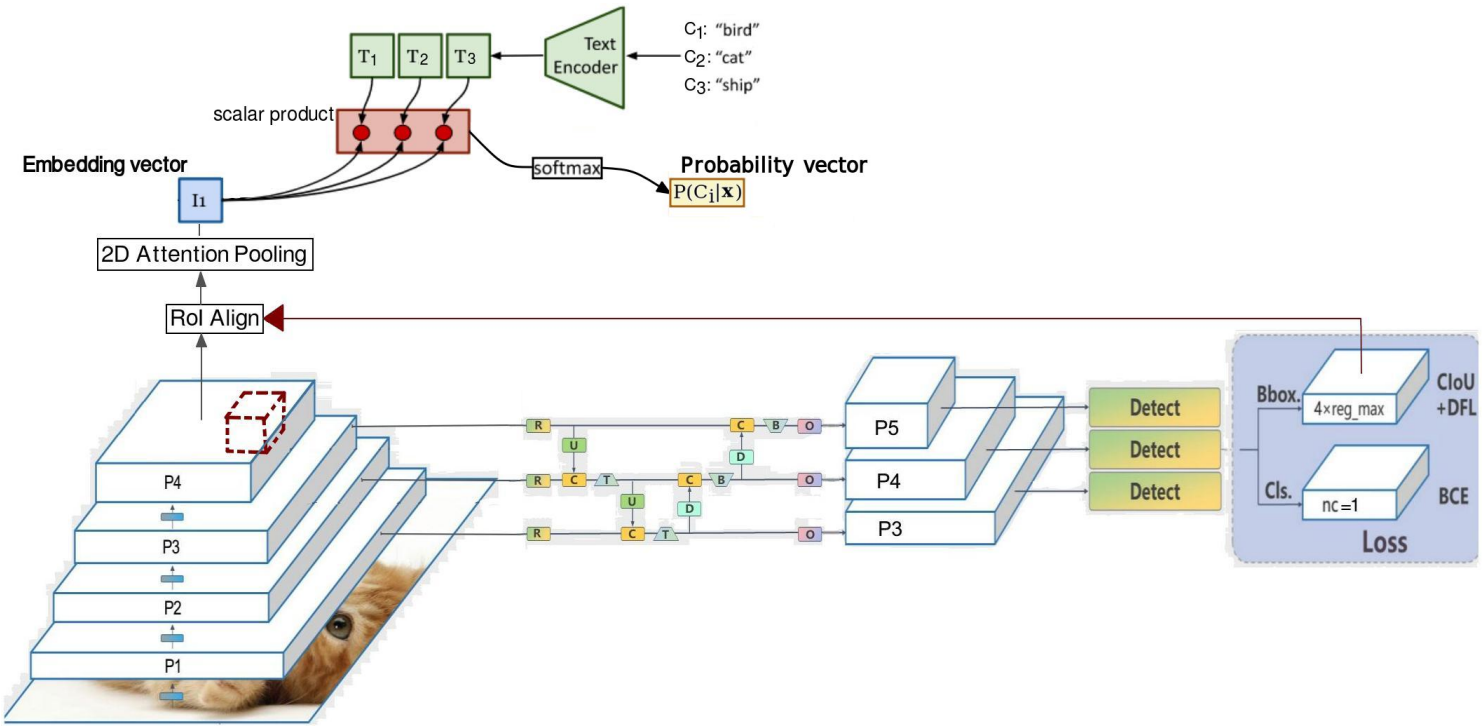


Figure III.6: YOLO-CLIP architecture

As a result, we obtain an open-vocabulary detector, we can calculate the text embedding for anything we want, not necessarily only classes, i.e "dog", "cat", "person", but adjectives, descriptions or scenes, i.e "a brown dog", "a person with stripped shirt", "a beautiful beach", etc.

As the possibilities for text expression are endless, a natural discussion into "prompt optimization" arrives, which looks to find the text prompt that gives best classification results. That depends on the data and the type of multi-modal pre-training the model had, in our case we used the CLIP encoders and CLIP was trained on image-caption pairs gathered from the web, the original paper observed that "a photo of a {label}." was a good default template and just using it increased CLIP's accuracy on ImageNet dataset by 1.3%. On fine-grained datasets, specifying the category also helped, i.e "A photo of a {label}, a type of pet.". Another technique is to ensemble many different prompts like "A photo of a {label}." and "A {label}", emsembling together with prompt engineering can increase the accuracy by almost 5% on ImageNet. [6]. In our case for detection on MSCOCO dataset we have used a single text prompt for each class: "A photo of a {label}."

It is important to emphasize that by using a RoI extractor, we've made our model a two-stage detector, since we now have one model responsible for localization of objects in the scene (class agnostic YOLO) and another model responsible for processing the many regions and classifying them.

Part IV

Experimentation

All the experimentation done in this section was made using solely CPU calculations at my desktop computer at XXII. The computer's specifications are:

- CPU Intel Xeon E5-2640 v3:
  - Nb. de cœurs 8
  - Nb. de threads 16
  - Fréquence Turbo maxi 3,40 GHz
  - Fréquence de la technologie Intel® Turbo Boost 2.0: 3.40 GHz
  - Fréquence de base 2,60 GHz
  - Cache 20 MB Intel® Smart Cache
- RAM Micron LRDIMM:
  - size: 32GiB
  - width: 64 bits
  - clock: 1866MHz (0.5ns)

IV.1 Class-agnostic Detector

We trained our class agnostic YOLO detector with pre-trained backbone (fig. III.3), following the procedures on section III.1.2, we'll reffer to it as YOLO-pretrained. It was trained on the MSCOCO2017 dataset [50], which is a dataset for object detection consisting of 80 categories, with 118K training images and 5k test images. We visualize the test results on image IV.1.

We compare the performance of our model with the original YOLOv8 large model (fig III.2). The YOLOv8 model was not trained in class agnostic fashion, it was trained to classify among the COCO classes. To compare it with the class agnostic YOLO-pretrained we unified the classification score given at the classification head to a single value. In a sense we are only evaluating the ability of a model to localize an object with a certain certainty and not to classify which object it is. We observe the results on table IV.3. AP stands for Average Precision and AR for Average Recall (precision and recall can be seen on equation IV.1).

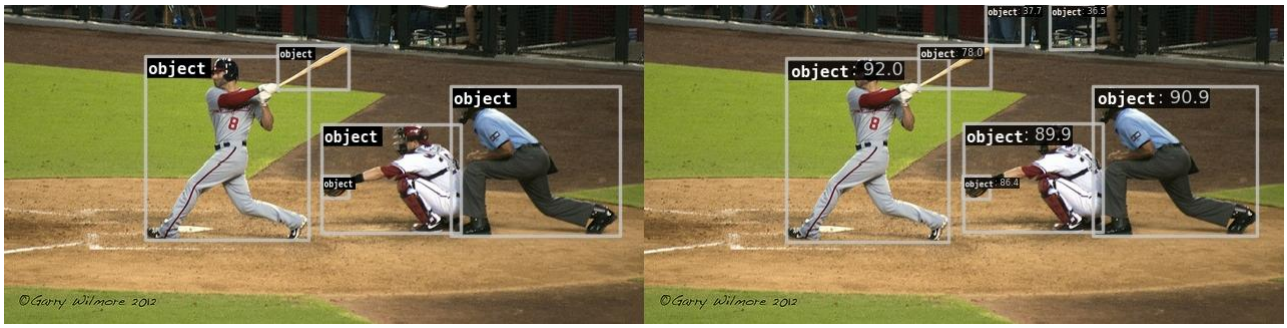


Figure IV.1: Ground-truth x Class agnostic detector inference results.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \qquad Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{IV.1}$$

	YOLOv8 large	**YOLO-pretrained**
Inference time (s/image)	0.0505	0.0985
AP[0.5:0.95]	0.547	0.358
AP[0.5]	0.75	0.531
AP[0.75]	0.594	0.379
AP[0.5:0.95] small	0.368	0.123
AP[0.5:0.95] medium	0.624	0.445
AP[0.5:0.95] large	0.762	0.657
AR[0.5:0.95]	0.12	0.11
AR[0.5]	0.496	0.374
AR[0.75]	0.686	0.431
AR[0.5:0.95] small	0.516	0.151
AR[0.5:0.95] medium	0.763	0.544
AR[0.5:0.95] large	0.875	0.765

Table IV.1: One-class localization performance of YOLOv8 large and our class agnostic YOLO with pre-trained backbone.

We observe that changing the backbone to a RN50 made our model take 95% more time. Our class agnostic model is also not as performing in detecting the objects as the original YOLOv8 on the MSCOCO. The recall and precision for small objects is the one that suffers the biggest difference, that is because the RN50 backbone was pretrained on fixed sized images of the size 224x224, which would be a medium-large sized object for the YOLO model that takes in images of size 640x640 during training, making the resolution of the backbone not adapted to smaller objects and not as performing in encoding features to distinguish it. Also we observe the recall has suffered a drop of performance of 8% compared to the drop in the precision of 34%. Looking at equations IV.1 this indicates that our model is making considerably more false positives, but the proportional amount of false negatives didn't change as much, this is expected from the class agnostic model since the objective is to classify any object and not only the 80 COCO classes. We consider that a high amount of false positives is not necessarily undesirable if it helps identifying real things on the images. Results can be seen on the Appendix IV.5.

IV.2 Zero-shot Object Detection

Zero-shot object detection is the task of object detection where no visual training data is available for some of the target object classes. A benchmark for zero-shot object detection was proposed by Bansal et al. [51], which presented a split of the MSCOCO dataset between 'base' and 'rare' categories (table IV.5) so that during training the model would see only the base categories and later be tested to identify both the base and rare ones. In the context of Open-vocabulary detection it refers to never having seen an annotated bounding box of the class of interest during training, this was formalized by Gu et al. [39] which also split the LVIS dataset into base and rare categories (table IV.5). Gu et al. [39] used a binary mask (that excluded rare classes) on the classification loss of the detector in order to train it, this way the model would not learn to classify the rare categories, although the regression head for localization remained unchanged and would still learn from the rare objects.

	COCO	**LVIS**
Number of base/rare categories	48/17	866/337

Table IV.2: Split of the dataset COCO for zero-shot object detection.

YOLO-CLIP model is made by coupling the class agnostic detector with an open vocabulary classification head as shown in section III.2, for this reason it was never trained on class labels of its objective classes and can be tested in zero-shot manner. To accelerate our deployment instead of encoding the text embeddings each time, we pre-calculated the text embeddings of the classes we wished to detect. We observe the results for test on COCO dataset on table IV.3.

YOLO-CLIP	mAP all (80)	mAP base(48)	mAP rare (17)
Inference time (s/image)	0.1082	0.108	0.108
AP[0.5:0.95]	0.156	0.143	0.203
AP[0.5]	0.239	0.225	0.309
AP[0.75]	0.165	0.152	0.216
AP[0.5:0.95] small	0.058	0.055	0.910
AP[0.5:0.95] medium	0.179	0.153	0.228
AP[0.5:0.95] large	0.257	0.243	0.286
AR[0.5:0.95]	0.183	0.171	0.240
AR[0.5]	0.318	0.324	0.451
AR[0.75]	0.333	0.344	0.471
AR[0.5:0.95] small	0.099	0.106	0.170
AR[0.5:0.95] medium	0.375	0.374	0.513
AR[0.5:0.95] large	0.543	0.568	0.663

Table IV.3: YOLO-CLIP results on zero-shot object detection on COCO dataset

By subtracting the inference speed of this with our class-agnostic model we can get the approximate time that the CLIP classification head takes, rougthly 0.1 seconds. Also the performance on classes considered rare on the dataset is higher than the base categories. Again we observe that small objects are the ones that suffer the most performance drop. We provide some visualization for some for the results of our model in figure IV.2 (more

results can be seen in the appendix IV.7). By looking at the results we observe some tendencies of the model, mainly related to detecting a person: the model is very bad at detecting people and this is also observed at the RegionCLIP model. Considering the training used by CLIP this is understandable because the image-caption pairs were taken from the internet, and the way we normally describe an image that includes people is by focusing on the activity they are performing or objects they carry, i.e. "Photo of a surfer on the sea", "Photo of a man with a backpack", so the CLIP model would learn to pay attention to words related to these activities like "skateboard", "backpack", "frisbee", "tennis racket", "cell phone" as they provide more discriminative information. For this reason the "person" category is more often identified with the categories related to activities and clothes: "skateboard", "backpack", "frisbee", "tennis racket", "cell phone" (this is shown on the confusion matrix at the appendix IV.6).

Figure IV.2: YOLO-CLIP inference results.

We can compare these results with the models discussed previously on the literature as seen in table IV.5 ($COCO_{-R}$ means the COCO dataset without the rare categories).

Model	Backbone	Pre-training	Detector	Detector training	Inference time (s/image)	COCO mAP
CLIP with GT boxes	RN50	CLIP [1]	-	-	-	0.583
RegionCLIP	RN50 [2]	CC3M	FasterRCNN	COCO	1.01	0.134 [5]
FVLM	RN50 [4]	-	FasterRCNN	LVIS	-	0.325
UniDetector	RN50 [3]	-	FasterRCNN	$COCO_{-R}$	0.323	0.307 [5]
UniDetector	RN50 [3]	-	FasterRCNN	$COCO_{-R}$ O365 OpenImages	0.323	0.512
YOLOCLIP	RN50 [4]	-	YOLO	$COCO_{-R}$	0.108	0.156

Table IV.4: Model comparison. [1]: data for the original CLIP model was never made available. [2]: RegionCLIP distilled knowledge from the original CLIP encoder and also used CC3M for pre-training. [3]: Unidetector uses the pre-trained backbone of RegionCLIP. [4]: FVLM and YOLOCLIP used CLIP's pretrained backbone. [5]: These results are not mentioned in the original paper but are direct results from running the provided codes using the provided checkpoints.

We observe that our model is by far the fastest among these and more performing

than the RegionCLIP trained on COCO even without performing grounding, having used CLIP backbone directly. Also, if we ask CLIP to classify the ground truth boxes, it obtains a much better performance than all the current OVD detectors on the literature, demonstrating that there's still much to be improved by increasing the detector's performance, this result is even better by giving RegionCLIP's encoder ground truth boxes to classify [7].

IV.3 One-shot Object Detection

Another test we made was one-shot object detection, detecting an object that the model has seen once before. For this we didn't use the text embeddings, what we did was to encode an image example of what we are looking for and compare the RoI embeddings to this encoded example on the embedding space directly. This is similar to do classification based on kNN, we make classification directly on the latent space of the image encoder, associating close instances to the same class. This was inspired by the fact that CLIP has been shown to have a well distinguished latent space, capable of separating instances from different classes (fig. IV.3). We tested the CLIP backbone on the task of image classification using ImageNet images from 48 classes that are common between ImageNet and COCO, obtained an accuracy of 82.8%.

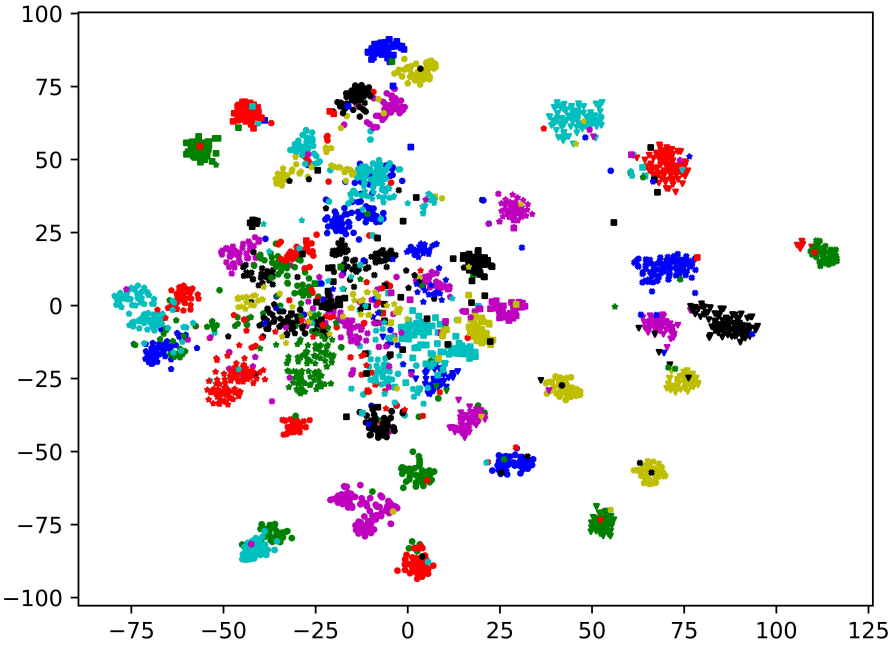


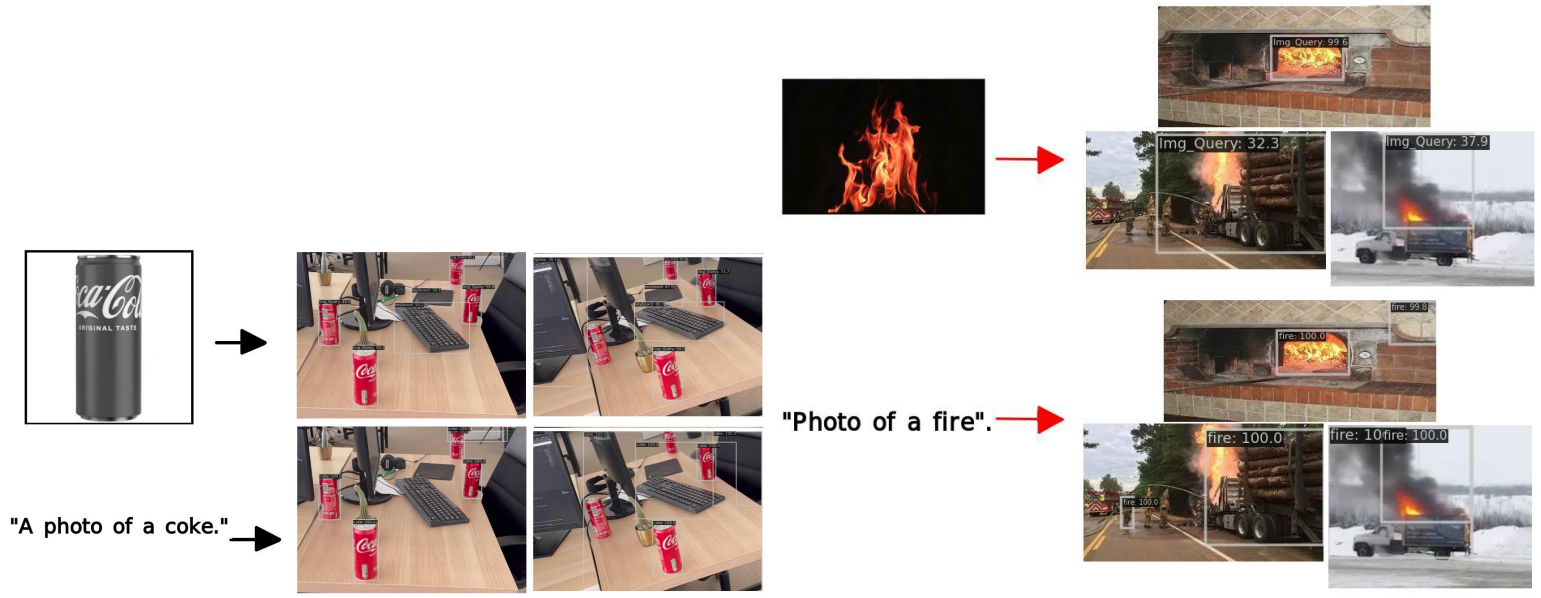
Figure IV.3: t-SNE visualization on image embeddings of CLIP. A different color and symbol is used for class of a total of 48 classes from COCO.

We have tested this approach for classification on the COCO dataset, since there are classes in COCO that are not in ImageNet we manually constructed an example dataset containing 1 image of each class, encoded each image example and during inference assigned each bbox to the class it was closest to in the embedding space. This would be like performing kNN classification for $k = 1$ and also 1 single instance for each class on the embedding space, a very challenging task. The results are given on table IV.5, the model's performance droped by 33% compared to the zero-shot setting.

	YOLO-CLIP
Inference time (s/image)	0.1082
AP[0.5:0.95]	0.102
AP[0.5]	0.149
AP[0.75]	0.108
AP[0.5:0.95] small	0.041
AP[0.5:0.95] medium	0.125
AP[0.5:0.95] large	0.155
AR[0.5:0.95]	0.128
AR[0.5]	0.218
AR[0.75]	0.229
AR[0.5:0.95] small	0.077
AR[0.5:0.95] medium	0.267
AR[0.5:0.95] large	0.343

Table IV.5: "One-shot" object detection on COCO dataset

These results are not necessarily good compared to the Open-vocabulary setting of comparing text to image, mainly because there's only one image example for each class. But what this approach excel is in image search, when we have an image example that is very close to the ones we are looking for in the image. We have made a test for object detection using a coke can as the image query of reference for the search as can be seen on image IV.4a, the model was tasked with predicting among the 80 COCO classes + an extra 'coke' class using only a single image example for each category. On image IV.4b we show only the n objects detected that are the closest to their respective category, we see that all of them consist of the image query (in this case fire) and not the other categories like 'truck' or 'oven', meaning that the model has a higher certainty of detecting objects when provided accurate image examples of more specific objects.



(a) One-shot coke image search. We used an image query in color and in black and white obtaining equally good results.

(b) One-shot fire search. We show the $n = 1$ objects closest to their respective image query.

Figure IV.4: One-shot image search from image query and text query.

We can see that given a good image example the model can correctly identify the object on the image and even give less false positives than text query search. Also when using image queries the model is less likely to wrongly assign labels to intersecting bounding boxes that actually contain different objects.

IV.4 Discussion

The class agnostic model with pretrained encoder has shown to have good localization performance compared to training a YOLOv8 large normally on the COCO dataset. The precision suffered a bigger drop than recall, meaning our model is making proportionally more false positives than false negatives, this is not necessarily a bad thing as the objective in deployment is making a model capable of detecting anomalies and many other categories not present in COCO dataset. Smaller objects have worse detection performance than medium and larger ones, this is common to other OVD models like RegionCLIP and UniDetector as they use RN50 backbone that takes in during pre-training middle-large size images.

YOLOCLIP model has shown to be by far the fastest model on the literature for OVD, 3x times faster than UniDetector the fastest one yet and 10x faster than RegionCLIP. In terms of performance it is not as performing as the current models in the literature like FVLM and UniDetector, but it is superior to RegionCLIP (2021) while having no need for grounding pre-training. A simple way to make it immediately more performing would be to do calibration as proposed by UniDetector. When looking at the results from UniDetector (table IV.5), which is the current SOTA using the RN50 backbone, we see that the performance of our model could be much improved training on different datasets.

Analyzing YOLO-CLIP's results on zero-shot open-vocabulary recognition IV.3 we see that the model has considerably higher performance on the rare categories than on base categories, showing that combining both class-agnostic localization and a frozen vision-language model can be beneficial for anomaly detection. We remember that combining both of these techniques is novel to the literature as UniDetector used a class agnostic RPN training but didn't freeze the backbone and the contrary can be said about F-VLM.

From table IV.5 we observe that training a better detector is crucial to increase performance, as given the GT boxes the CLIP model can obtain much better classification results than even current SOTA. The backbone used for our model was pre-trained to do image-caption alignment, this is not the same task performed by an OVD which requires to align specific regions of the image with text. Using a backbone trained on region-text alignment or grounding might also provide superior performance.

For these reasons we tried to train our class-agnostic detector using RegionCLIP's pre-trained encoder on the LVIS dataset that contains much more classes and annotations. During the class agnostic pre-training we observed repeated errors by CUDA GPU calculations, specifically in the loss calculation part of the code responsible for matching each proposed bbox to a specific ground truth box. That is because LVIS has much more annotations per image and also because the class-agnostic training requires us to unify all the annotation's class label, so the matching algorithm can't use class prediction information to narrow down the search during the matching between GT bboxes and proposed bboxes. This caused the programme to crash repeatedly either for lack of GPU memory available or for the GPU cores becoming unsinchronized with each other.

We observed by looking at the predicted results that YOLO-CLIP has a limitation which comes from the CLIP model itself. If the RoI of object 1 is contained inside the RoI

of object 2, the model might predict object 1 from object's 2 bbox. Since CLIP is used to looking at a whole image and understanding it in context it might find some smaller detail more relevant than the bigger object. This is however less likely to happen if we do one-shot detection using an image query instead of a text query.

Also a particularly interesting application has been studied through one-shot object detection, where we classify objects from a given image prompt. Because of the fact the pre-trained latent space of the CLIP model is well-structured and adapted to image classification, classification can be done directly on it by nearest-neighbors. This has been shown to provide good results if the image query is close to the object of interest. This approach allows YOLO-CLIP to not only do search from text but from an image example as well.

Conclusion

We now return to the context and prospectives defined on the beginning of this study (section I). Two main prospectives for this internship: reducing the cost of collecting and labeling data of rare classes and performing real-time anomaly detection. We consider to have answered to both, our YOLO-CLIP model is capable of real-time object detection and also has shown promissing results on both text-to-image and image-to-image search.

To address these challenges we set out first to explore the current SOTA on the field of Out-Of-Distribution (OOD) and then on the field of Vision-Language Models (VLM), finally choosing to adress it through Open Vocabulary Object Detection (OVD).

On a personal note the internship was incredibly fruitful as I explored many relevant fields on computer vision, obtained a very solid base on software architecture design, on different machine learning and data analysis frameworks, as well as software development tools like Docker, VScode, GitHub.

IV.4.1 Contributions

We cite our main contribution as proposing for the first time in the literature a novel model adapted for real-time OVD. Our approach uses a frozen pre-trained backbone, maintaining the relevant features learned during training making it more adapted for detecting rare classes. We provide a regime to train a single stage detector to do class-agnostic object detection making it suitable for the OVD setting, until now OVD models in the literature all have used a two stage detector architecture. As secondary contributions we cite doing a comprehensive overview of the settings and approaches on the fields of Out-Of-Distribution (OOD), Vision-Language Models (VLM) as well as performing a SOTA on the field of Open Vocabulary Object Detection (OVD).

IV.4.2 Future works

A lot of work can be done to improve the performance and generality of our approach. Training the model on bigger datasets like LVIS and Object365, either separetely or jointly like proposed on the UniDetector paper [9]. Also using a backbone trained on the task of region-text alignment instead of image-text alignment. A simple calibration wasn't done for lack of time but could have increased the performance of our model.

A totally different and exciting approach would be to use transformers, currently there transformer architectures capable of real-time object detection [52]. Although there aren't transformer models on these architectures pre-trained through contrastive learning and consequently that could be used for OVD, approaches like distillation could allow one to transfer the knowledge from one transformer incapable or real-time object detection to another one that is.

Bibliography

[1] Colin Troisemaine, Vincent Lemaire, Stéphane Gosselin, Alexandre Reiffers-Masson, Joachim Flocon-Cholet, and Sandrine Vaton. Novel Class Discovery: an Introduction and Key Concepts, February 2023. URL http://arxiv.org/abs/2302.12028. arXiv:2302.12028 [cs].

[2] Relja Arandjelović, Alex Andonian, Arthur Mensch, Olivier J. Hénaff, Jean-Baptiste Alayrac, and Andrew Zisserman. Three ways to improve feature alignment for open vocabulary detection, March 2023. URL http://arxiv.org/abs/2303.13518. arXiv:2303.13518 [cs].

[3] Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized Out-of-Distribution Detection: A Survey, August 2022. URL http://arxiv.org/abs/2110.11334. arXiv:2110.11334 [cs].

[4] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks, August 2019. URL http://arxiv.org/abs/1908.02265. arXiv:1908.02265 [cs].

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, June 2021. URL http://arxiv.org/abs/2010.11929. arXiv:2010.11929 [cs].

[6] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision, February 2021. URL http://arxiv.org/abs/2103.00020. arXiv:2103.00020 [cs].

[7] Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, and Jianfeng Gao. RegionCLIP: Region-based Language-Image Pretraining, December 2021. URL http://arxiv.org/abs/2112.09106. arXiv:2112.09106 [cs].

[8] Weicheng Kuo, Yin Cui, Xiuye Gu, AJ Piergiovanni, and Anelia Angelova. F-VLM: OPEN-VOCABULARY OBJECT DETECTION UPON FROZEN VISION AND LANGUAGE MODELS. 2023.

[9] Zhenyu Wang, Yali Li, Xi Chen, Ser-Nam Lim, Antonio Torralba, Hengshuang Zhao, and Shengjin Wang. Detecting Everything in the Open World: Towards Uni-

versal Object Detection, March 2023. URL http://arxiv.org/abs/2303.11749. arXiv:2303.11749 [cs].

[10] MMDetection Contributors. OpenMMLab Detection Toolbox and Benchmark, August 2018. URL https://github.com/open-mmlab/mmdetection. original-date: 2018-08-22T07:06:06Z.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, July 2023. URL http://arxiv.org/abs/1706.03762. arXiv:1706.03762 [cs].

[12] Jamie Murdoch. How I found nearly 300,000 errors in MS COCO, July 2022. URL https://medium.com/@jamie_34747/how-i-found-nearly-300-000-errors-in-ms-coco-79d382edf22b.

[13] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from Noisy Labels with Deep Neural Networks: A Survey, March 2022. URL http://arxiv.org/abs/2007.08199. arXiv:2007.08199 [cs, stat].

[14] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. FLAVA: A Foundational Language And Vision Alignment Model. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15617–15629, New Orleans, LA, USA, June 2022. IEEE. ISBN 978-1-66546-946-3. doi: 10.1109/CVPR52688.2022.01519. URL https://ieeexplore.ieee.org/document/9880206/.

[15] Stanislav Fort, Jie Ren, and Balaji Lakshminarayanan. Exploring the Limits of Out-of-Distribution Detection, July 2021. URL http://arxiv.org/abs/2106.03004. arXiv:2106.03004 [cs].

[16] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. CutPaste: Self-Supervised Learning for Anomaly Detection and Localization, April 2021. URL http://arxiv.org/abs/2104.04015. arXiv:2104.04015 [cs].

[17] Guangyao Chen, Peixi Peng, Xiangqian Wang, and Yonghong Tian. Adversarial Reciprocal Points Learning for Open Set Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. ISSN 0162-8828, 2160-9292, 1939-3539. doi: 10.1109/TPAMI.2021.3106743. URL http://arxiv.org/abs/2103.00953. arXiv:2103.00953 [cs].

[18] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4393–4402. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/ruff18a.html.

[19] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent Advances in Open Set Recognition: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(10):3614–3631, October 2021. ISSN 0162-8828, 2160-9292, 1939-3539. doi: 10.1109/TPAMI.2020.2981604. URL http://arxiv.org/abs/1811.08581. arXiv:1811.08581 [cs, stat].

[20] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Open-Set Recognition: a Good Closed-Set Classifier is All You Need?, April 2022. URL http://arxiv.org/abs/2110.06207. arXiv:2110.06207 [cs].

[21] Jing Lu, Yunxu Xu, Hao Li, Zhanzhan Cheng, and Yi Niu. PMAL: Open Set Recognition via Robust Prototype Mining, March 2022. URL http://arxiv.org/abs/2203.08569. arXiv:2203.08569 [cs].

[22] Yiyou Sun, Chuan Guo, and Yixuan Li. ReAct: Out-of-distribution Detection With Rectified Activations, November 2021. URL http://arxiv.org/abs/2111.12797. arXiv:2111.12797 [cs].

[23] Yu Du, Fangyun Wei, Zihe Zhang, Miaojing Shi, Yue Gao, and Guoqi Li. Learning to Prompt for Open-Vocabulary Object Detection with Vision-Language Model. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14064–14073, New Orleans, LA, USA, June 2022. IEEE. ISBN 978-1-66546-946-3. doi: 10.1109/CVPR52688.2022.01369. URL https://ieeexplore.ieee.org/document/9878606/.

[24] Haoran Wang, Weitang Liu, Alex Bocchieri, and Yixuan Li. Can multi-label classification networks know what they don't know?, October 2021. URL http://arxiv.org/abs/2109.14162. arXiv:2109.14162 [cs].

[25] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.

[26] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Generalized Category Discovery, June 2022. URL http://arxiv.org/abs/2201.02609. arXiv:2201.02609 [cs].

[27] Kaidi Cao, Maria Brbic, and Jure Leskovec. Open-World Semi-Supervised Learning, January 2022. URL http://arxiv.org/abs/2102.03526. arXiv:2102.03526 [cs].

[28] Wenbin An, Feng Tian, Qinghua Zheng, Wei Ding, QianYing Wang, and Ping Chen. Generalized Category Discovery with Decoupled Prototypical Network, March 2023. URL http://arxiv.org/abs/2211.15115. arXiv:2211.15115 [cs].

[29] Sheng Zhang, Salman Khan, Zhiqiang Shen, Muzammal Naseer, Guangyi Chen, and Fahad Khan. PromptCAL: Contrastive Affinity Learning via Auxiliary Prompts for Generalized Novel Category Discovery, March 2023. URL http://arxiv.org/abs/2212.05590. arXiv:2212.05590 [cs].

[30] Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. Vision-Language Models for Vision Tasks: A Survey, April 2023. URL http://arxiv.org/abs/2304.00685. arXiv:2304.00685 [cs].

[31] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. VisualBERT: A Simple and Performant Baseline for Vision and Language, August 2019. URL http://arxiv.org/abs/1908.03557. arXiv:1908.03557 [cs].

[32] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. URL http://arxiv.org/abs/1810.04805. arXiv:1810.04805 [cs].

[33] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D. Manning, and Curtis P. Langlotz. Contrastive Learning of Medical Visual Representations from Paired Images and Text, September 2022. URL http://arxiv.org/abs/2010.00747. arXiv:2010.00747 [cs].

[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, December 2015. URL http://arxiv.org/abs/1512.03385. arXiv:1512.03385 [cs].

[35] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision, June 2021. URL http://arxiv.org/abs/2102.05918. arXiv:2102.05918 [cs].

[36] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset For Automatic Image Captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia, 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1238. URL http://aclweb.org/anthology/P18-1238.

[37] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. SimVLM: Simple Visual Language Model Pretraining with Weak Supervision, May 2022. URL http://arxiv.org/abs/2108.10904. arXiv:2108.10904 [cs].

[38] Alireza Zareian, Kevin Dela Rosa, Derek Hao Hu, and Shih-Fu Chang. Open-Vocabulary Object Detection Using Captions, March 2021. URL http://arxiv.org/abs/2011.10678. arXiv:2011.10678 [cs].

[39] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary Object Detection via Vision and Language Knowledge Distillation, May 2022. URL http://arxiv.org/abs/2104.13921. arXiv:2104.13921 [cs].

[40] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, Kai-Wei Chang, and Jianfeng Gao. Grounded Language-Image Pre-training, June 2022. URL http://arxiv.org/abs/2112.03857. arXiv:2112.03857 [cs].

[41] Haotian Zhang, Pengchuan Zhang, Xiaowei Hu, Yen-Chun Chen, Liunian Harold Li, Xiyang Dai, Lijuan Wang, Lu Yuan, Jenq-Neng Hwang, and Jianfeng Gao. GLIPv2: Unifying Localization and Vision-Language Understanding, October 2022. URL http://arxiv.org/abs/2206.05836. arXiv:2206.05836 [cs].

[42] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. Simple Open-Vocabulary Object Detection with Vision Transformers.

[43] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, January 2016. URL http://arxiv.org/abs/1506.01497. arXiv:1506.01497 [cs].

[44] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting Twenty-thousand Classes using Image-level Supervision, July 2022. URL http://arxiv.org/abs/2201.02605. arXiv:2201.02605 [cs].

[45] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution, February 2022. URL http://arxiv.org/abs/2202.10054. arXiv:2202.10054 [cs].

[46] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection, May 2016. URL http://arxiv.org/abs/1506.02640. arXiv:1506.02640 [cs].

[47] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. YOLO by Ultralytics, January 2023. URL https://github.com/ultralytics/ultralytics. original-date: 2022-09-11T16:39:45Z.

[48] Dillon Reis, Jordan Kupec, Jacqueline Hong, and Ahmad Daoudi. Real-Time Flying Object Detection with YOLOv8, May 2023. URL http://arxiv.org/abs/2305.09972. arXiv:2305.09972 [cs].

[49] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond Empirical Risk Minimization, April 2018. URL http://arxiv.org/abs/1710.09412. arXiv:1710.09412 [cs, stat].

[50] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context, February 2015. URL http://arxiv.org/abs/1405.0312. arXiv:1405.0312 [cs].

[51] Ankan Bansal, Karan Sikka, Gaurav Sharma, Rama Chellappa, and Ajay Divakaran. Zero-Shot Object Detection, July 2018. URL http://arxiv.org/abs/1804.04340. arXiv:1804.04340 [cs].

[52] Wenyu Lv, Yian Zhao, Shangliang Xu, Jinman Wei, Guanzhong Wang, Cheng Cui, Yuning Du, Qingqing Dang, and Yi Liu. DETRs Beat YOLOs on Real-time Object Detection, July 2023. URL http://arxiv.org/abs/2304.08069. arXiv:2304.08069 [cs].

Appendix

IV.4.3 Class-agnostic Detector

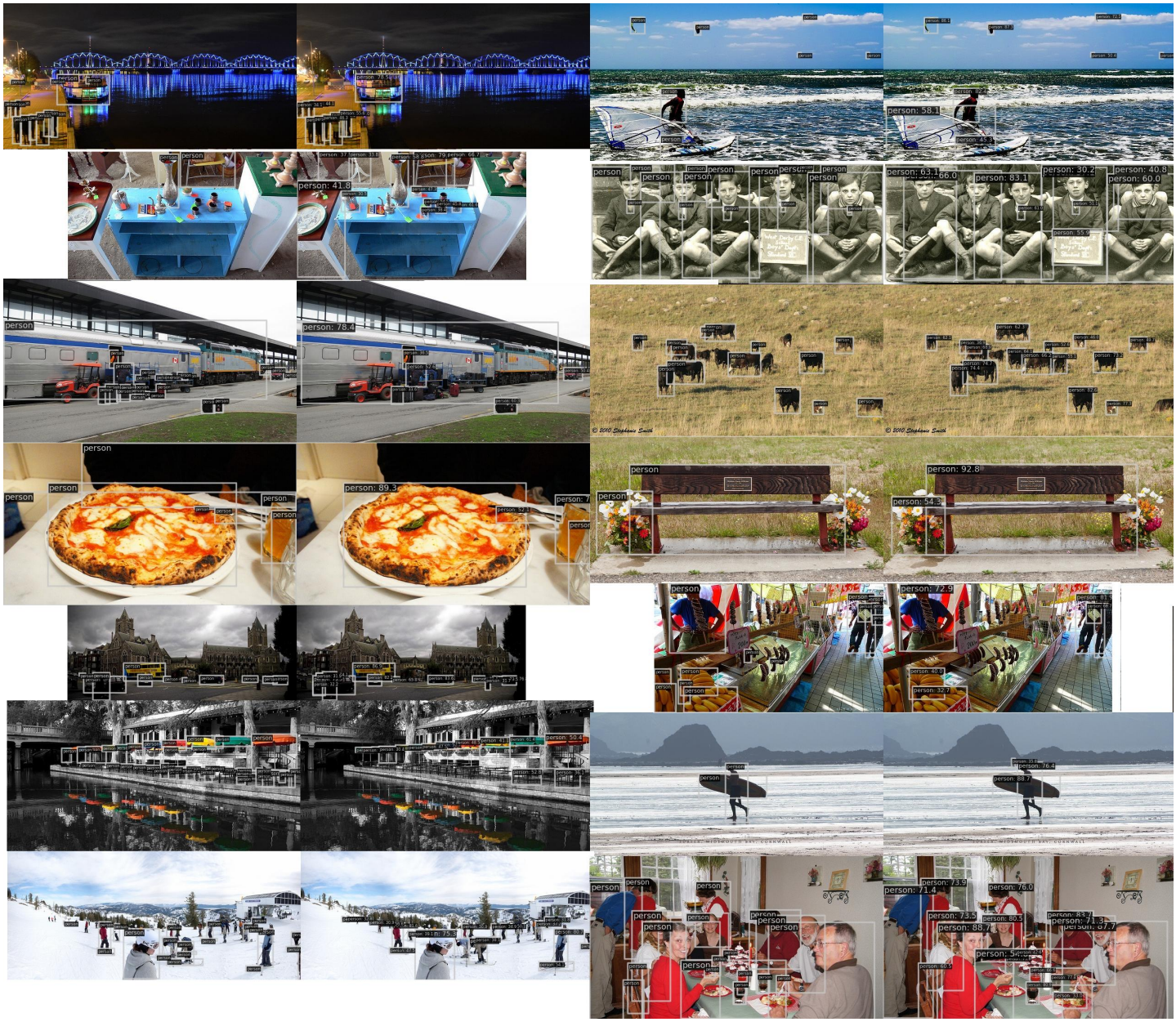


Figure IV.5: Ground-truth vs Class-agnostic detection. For visualization the annotation labels were unified to 0, which made every object to be exhibited as the first class 'person'.

IV.4.4 Zero-shot Object Detection

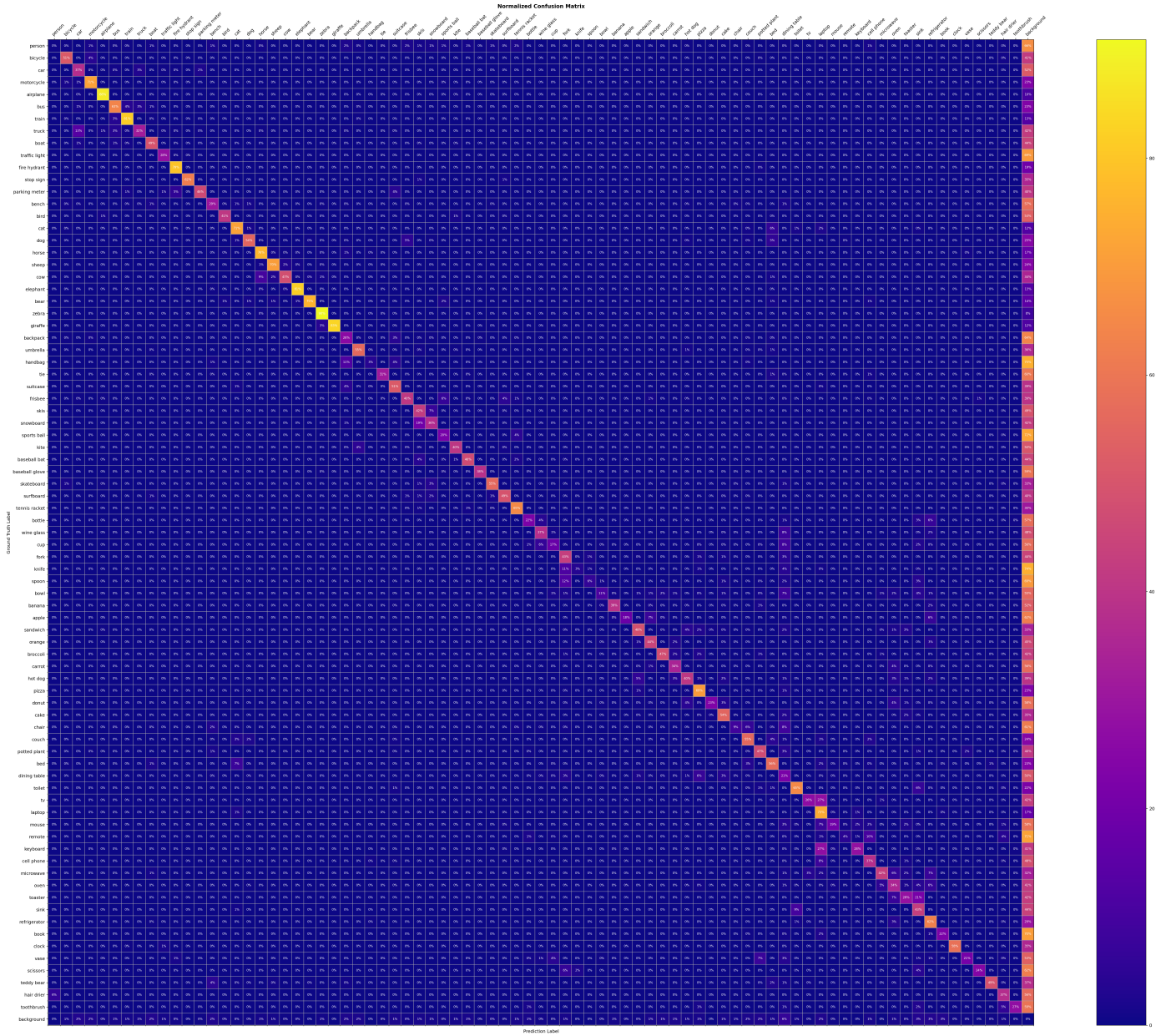


Figure IV.6: Confusion matrix of YOLO-CLIP model. Better seen digitally.

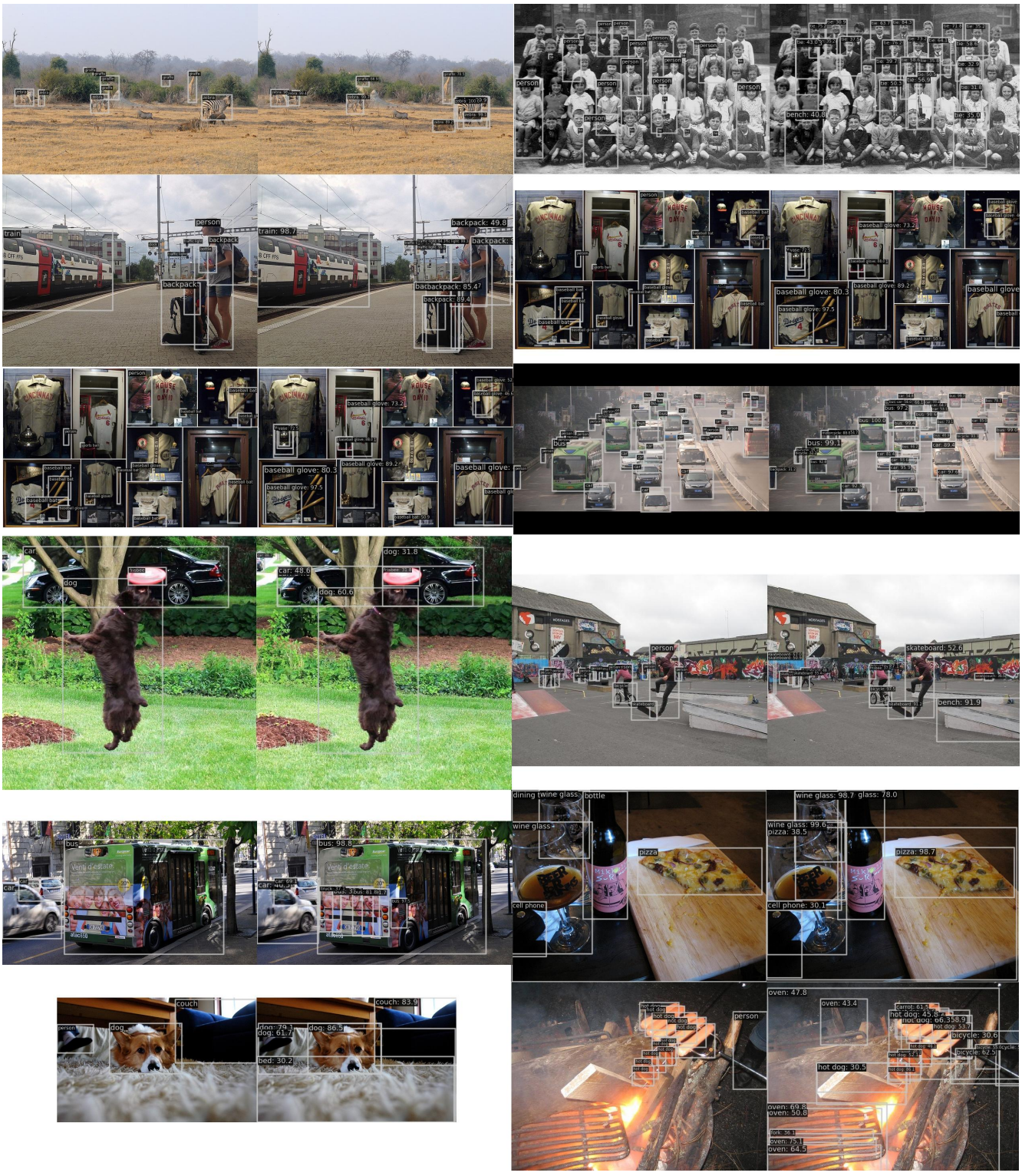


Figure IV.7: Ground-truth vs Zero-shot object detection with YOLO-CLIP.

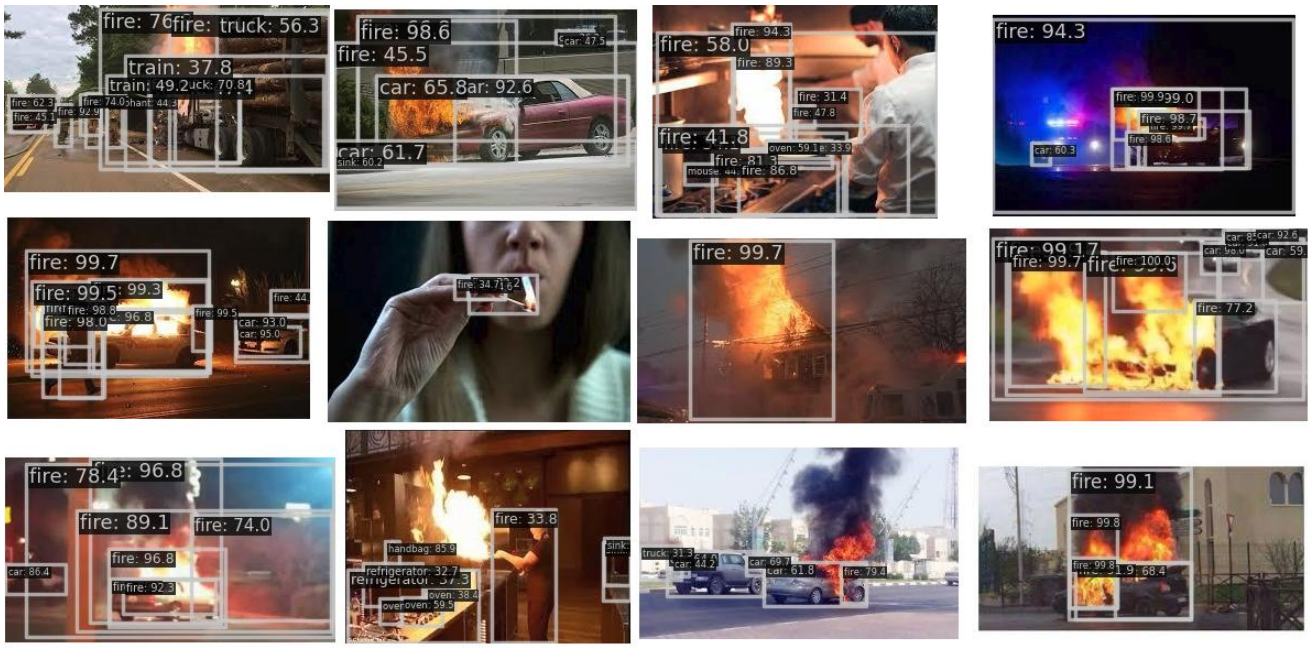


Figure IV.8: YOLO-CLIP zero-shot detection of all COCO categories + 'fire' category.

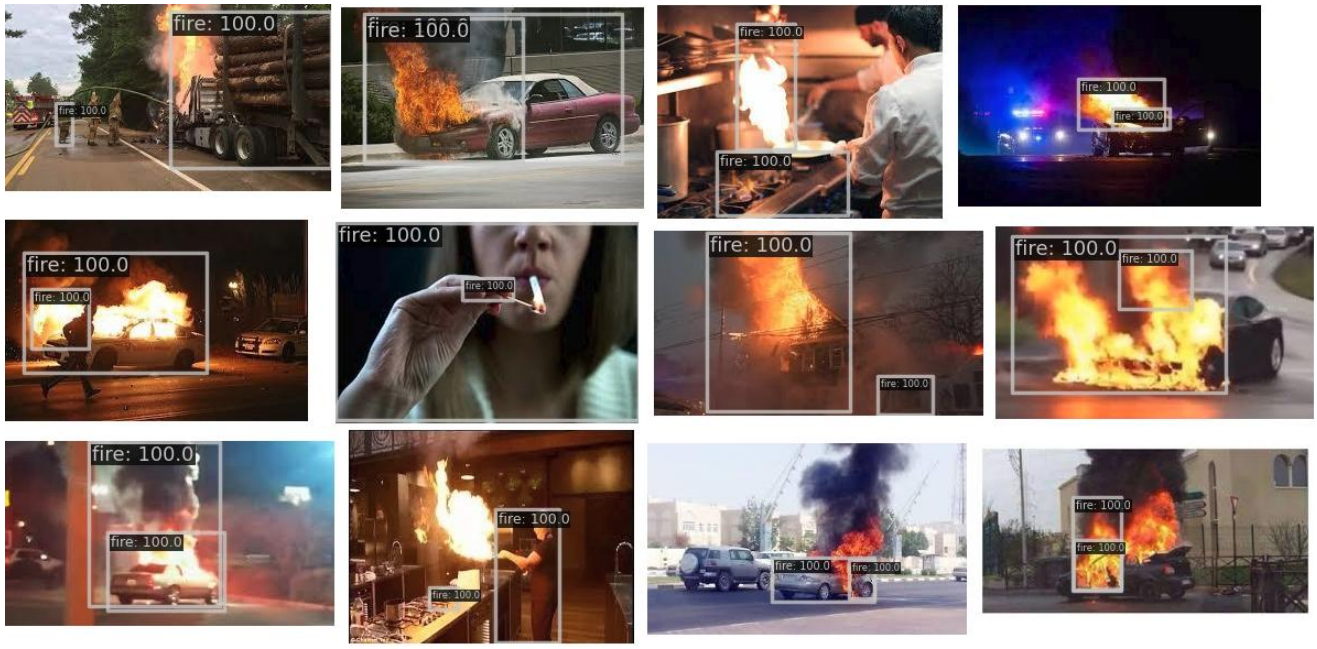


Figure IV.9: YOLO-CLIP zero-shot detection of all COCO categories + 'fire' category but only the n=2 objects classifications with highest certainty

Glossary

AD Anomaly Detection

CNN Convolutional Neural Network

DNN Deep Neural Network

FPN Feature Pyramid Network

GCD Generalized Category Discovery

ID In-Distribution

ML Machine Learning

NCD Novel Category Discovery

ND Novelty Detection

OCC One-Class Classification

OD Outlier Detection

OOD Out-Of-Distribution

OSR Open-Set Recognition

OVD Open Vocabulary Object Detection

PAN Path Aggregation Network

RNN Recurrent Neural Network

RoI Region of Interest

RPN Region Proposal Network

VLM Vision-Language Models