TUM Data Innovation Lab

Munich Data Science Institute

Technical University of Munich

&

PreciBake GmbH

Final report of project:

Detecting Novel Objects with only Few Examples

Authors	Pinar Ayaz, Ulrika Bremberg, Mariia Koren, Rohan Menon and Anna-Maria Weber
Mentor(s)	M.Sc. Mathias Sundholm, M.Sc. Alexander Dolokov, M.Sc. Maximilian Schreil, M.Sc. Sebastian Freytag
Project Lead	Dr. Ricardo Acevedo Cabra
Supervisor	Prof. Dr. Massimo Fornasier

Aug 2022

Abstract

Computer vision has grown tremendously over the past few years, and is now widely used to simplify and increase the efficiency of many industries. Especially object detection, a subfield of computer vision, has seen terrific growth over the past decade. Nevertheless, most techniques still have a long way to go before they can bridge the gap between human and computer vision.

Object detection advances can greatly benefit companies needing real-time inventory monitoring. However, in order to train object detectors with a sufficient level of accuracy, large datasets are required. Industry inventories typically contain hundreds or thousands of unique objects, making it challenging to create and maintain high quality datasets.

As a solution to this problem, we propose a pipeline for detecting novel objects in an image using only one query image as feature descriptor. As opposed to other state-of-the-art methods in few-shot and zero-shot detection, we present an approach that utilizes CLIP image embeddings as its unique feature. The pipeline includes an object localization network, a matching component that compares the CLIP embeddings of detected objects with the query image, and a captioning model that generates captions for the detected bounding boxes. We demonstrate the model's performance on the highly challenging Oktoberfest Food Dataset and Object365 Dataset, performing numerous tests illustrating the importance of choosing relevant query images and object detection architectures. Through comprehensive experiments we demonstrate how our approach of using CLIP embedding matching, along with few- and zero-shot object detection, can replace expensive datasets with promising results.

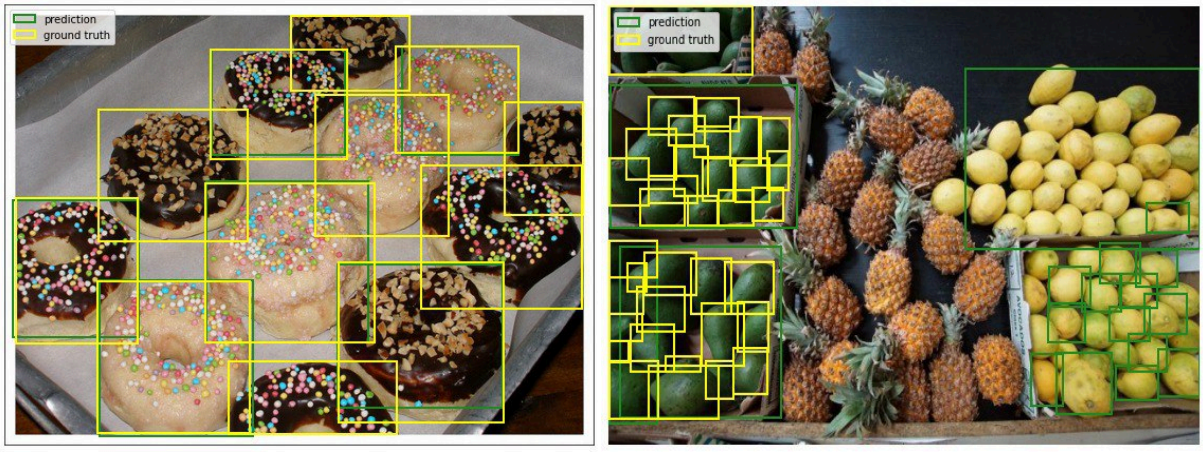
Contents

1 Introduction

Nowadays, the field of computer vision is not only a part of research, but also an important part of many areas in industry. This project is done in cooperation with PreciTaste, a company working on AI solutions for the food-tech and baking industry, developing machine learning models for image classification tasks, object detection and object tracking. For instance, they provide a real time inventory monitoring system [1], designed for use in bakery and restaurant settings.

The inventories in these environments might contain hundreds or thousands of unique products, which in turn makes it difficult to construct relevant, high quality datasets. The data itself can also change over time, for example with new packaging, or new items that need to be added to the datasets. Therefore, these scenarios would greatly benefit from recent advances in object detection. Consider the scenario visualized in Figure 1. It succinctly shows the problem which detectors face when they encounter a novel class. In this example, we used the object localization network DETR[2] to generate bounding boxes predictions of the food items in the images, which we then compare with the ground truth bounding boxes. The DETR model uses a ResNet model [3] pretrained on the COCO dataset [4], a dataset which contains a *Donuts* as a class whereas the *Avocados* class in Figure 1b is novel. It is easier for the model to detect donuts since it has seen them during training before. In contrast, the model does not recognize any of the avocados as objects, as these classes are unfamiliar to the model.

Models should not only detect and classify specific objects, but also be able to generalize well to new classes, as shown in the Figure 1. We approach this issue by handing the model a query image that can act as a feature descriptor of the object we are searching for in a reference image. For example, given an image of a donut, the model should be able to detect all donuts in a dataset matching the feature description, even when it was not trained for this specific task.



(a) (b)

Figure 1: Raw Bounding Box predictions for (a) COCO vs (b) non-COCO class performed using DETR[2] on Objects365 Dataset[5]

The goal of our project is to develop a universal object detection model that could detect any type of object in an image at test time, given just one sample of the object of interest. Our contributions demonstrate how CLIP embedding matching can be used along with few- and zero-shot object detection to replace expensive datasets, with notable results.

In Section 2, we give a short overview of the theoretical background and existing work in the areas of object detection, few-shot learning and zero-shot learning. Next, we present our proposed pipeline and describe our own approach which is based on the idea of utilizing CLIP image embeddings to perform a matching between the reference sample and all objects in an input image in Section 3. Furthermore, we discuss our experiments and results and present a comparison of the two types of object detection networks DETR [2] and OS2D [6] we used in our pipeline in Section 4. Eventually, in Section 5 we describe our ideas for possible improvements of the proposed model and what approaches are worth exploring in the future shortly. Finally, we summarize our findings in section 6.

2 Related Work

Object Detection. Object detection is one of the main tasks in computer vision, with the goal of detecting and classifying instances of objects within images. This task, of precisely estimating the concept and locations of objects in an image, is further divided into subtasks such as face detection and object tracking. Some early famous detection algorithms include Viola Jones Detection [7] (2001) or Histogram of Oriented Gradients [8] (2005). The transition towards using Convolutional Neural Networks (CNN) as backbone architectures in object detection came with the introduction of Regions with CNN features (R-CNN) [9] [10]. Since then, improved models have been suggested, some of the most famous being Fast R-CNN [11] introducing innovations that increase the train and test speed while also improving detection accuracy, Faster R-CNN [12] that use an additional Region Proposal Network (RPN), and YOLO [13] that frame object detection as a regression problem.

Modern object detectors can be divided into two categories: proposal-based (two-stage) and proposal-free (one-stage) networks. Both Fast R-CNN and Faster R-CNN belong to the proposal-based networks. These networks first pass the images through a feature extraction module, that is fed to a region proposal extractor generating bounding boxes. The region proposals are then refined and classified into different categories. In contrast, the one-stage detector networks do not have the region proposal extractor, but instead make the class prediction and bounding box coordinates in a single step. YOLO is an example of such a proposal-free network, using a single convolutional network to both perform class and bounding box predictions. Other single stage detectors include for example SSD [14] and RefineDet [15]. Historically, the one-stage detector networks have been more efficient, whereas the two-stage methods have achieved higher accuracy [16] [17].

Common benchmarks in object detection tasks are the MS COCO [18], LVIS [19], or CIFAR-10 [20] datasets.

Few-shot Learning. The vast improvements for using CNNs for object detection has been partly dependent on training on extensive datasets, which can lead to over-fitted models that may fail to generalize [21]. A solution to avoid these big and costly datasets is to let the machine learning model learn to classify and/or detect objects of new classes based on a few examples. This area of computer vision is called few-shot learning and aims to mimic how humans learn to distinguish new objects quickly by examining a set of very few examples [21].

The research has earlier been focused on few-shot classification, which is an easier task than few-shot object detection (FSOD). FSOD models requires both recognition of the object types, as well as localization of the object among millions of potential regions [22]. This implies that the classification methods cannot be directly transferred to solve the few-shot detection problem. FSOD experiments can be divided into meta learning and transfer learning approaches. In transfer learning, the final layers of a pretrained model are finetuned with unseen classes while the rest of the model weights are kept frozen. Some of these finetuning attempts include TFA [22], FSCE [23], and DeFRCN [24].

In contrast to transfer learning, the increasingly popular meta learning approaches aim to acquire task-level meta knowledge that then can help the model quickly adapt to new tasks and environments given very few novel, labeled, examples. Examples include Meta-DETR [25], performing meta learning on image level while capturing inter-class correlations among different support images, Meta-RCNN [26] extending Faster/Mask R-CNN with meta learning, and Few-shot Object Detection via Feature Reweighting [21] which combines a meta feature learner with a reweighting module.

Zero-shot Learning. The goal of zero-shot learning is to recognize objects that have not been seen during training. In contrast to few-shot learning, zero shot learners do not perform any fine tuning on novel classes. Instead, the knowledge learned in the training set is intended to be transformed into the ability of classifying the testing set [27]. Just like in few-shot object detection, it is more difficult to both detect and classify novel images than it is to only perform image classification. However, due to its great usability, zero-shot detection is a fast developing field in machine learning, and the number of proposed methods has been increasing rapidly over the last few years [28] [27]. Furthermore, zero-shot detection is a closer step to mimic the human cognition of the world. The fact that it is not dataset dependent makes it, potentially, very powerful in critical situations, such as obstacle detection in autonomous driving.

Some type of semantic information is often used as auxiliary information in zero-shot detection tasks. After training the models on labeled data, it is possible to help the model identify new objects by explaining the appearance in text [27]. An example could be to describe a leopard as a yellow-golden cat with dark spots to enable detection of it.

Recent attempts to tackle the zero-shot detector task include OLN [29] introducing a classifier-free object proposer, DETReg [30] introducing a model that learns to both localize and encode object simultaneously during the pretraining stage, and ViLD [31] distilling the knowledge from a pretrained open-vocabulary image classification model to enable open-vocabulary detection.

Image-text models. An increasingly popular approach to solve the need for extensive image datasets for object detection and classification is to make use of the vast paired image-text data accessible on the internet. Recent work such as CLIP [32] and ALIGN [33] have trained image-text models using contrastive learning. More specifically, CLIP is trained on 400 million image-text pairs with a wide variety of images and natural language supervision, and demonstrates impressive results transferring to over 30 datasets. The network is instructed in natural language to perform classification on a vast number of datasets, without the need of directly optimizing for any new data. Further extensions of CLIP are for example the previously mentioned ViLD model [31], and ZOC, a model which compares the semantic meaning of an image to both its previously seen labels and to model-generated candidate labels [34].

3 Methods and Pipeline

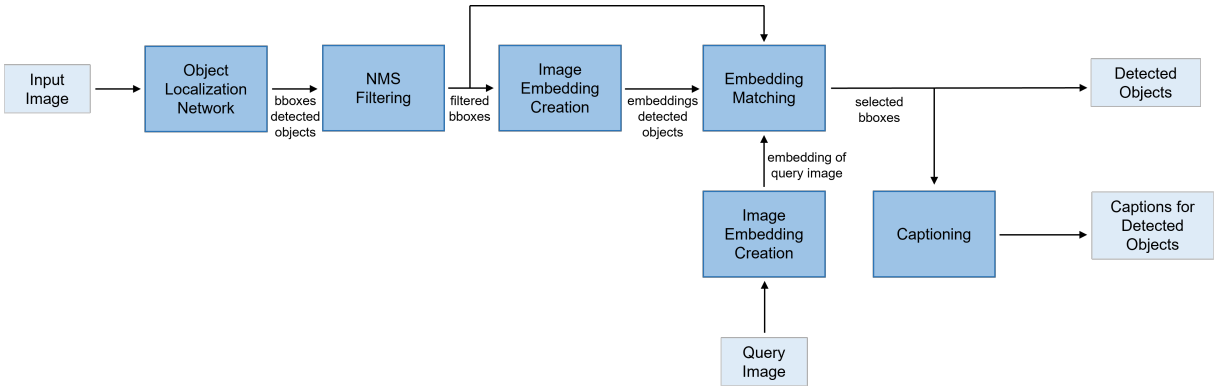


Figure 2: Proposed Pipeline

We developed a pipeline to detect novel objects in an input sample given only one example image, which we will refer to as query image. Different to the models presented in section 2, we propose an approach whose unique feature is the usage of CLIP image embeddings. The pipeline consists of several consecutive components: an object localization network followed by a non-maximum suppression step to predict high-quality bounding boxes, a matching component where the image embeddings of all predicted bounding boxes and the query image are compared and matched, and finally, a captioning model which creates captions for the selected bounding boxes. In the following sections, we will explain and discuss all components in detail.

3.1 Object Localization

The first component of our pipeline is a region proposal network. The generation of bounding box candidates is crucial, since only the predicted bounding boxes will be considered in the next steps of the pipeline. Optimally, the object localization model would provide

perfect bounding boxes for each object in an input image while ignoring the background. However, in reality also poor and wrong bounding boxes are going to be predicted.

Since the localization of objects is a significant part of object detection, we put a lot of emphasis on finding a suitable region proposal network. Searching for a robust and efficient model, we reviewed multiple detection models, including zero-shot, one-shot and few-shot detection networks. In this process, we also explored models using meta learning, a completely different approach to the problem. At all times, we focused on models that do not depend on text inputs, since our goal was to create a pipeline using images only.

3.1.1 Zero-Shot detection

The first method we explored was zero-shot detection. A zero-shot detection model returns bounding boxes for all objects in an input image without receiving any further information. This approach is highly challenging, following from the fact that it is an insoluble task to define precisely what an object is and what is not. Thinking of a house, the object could be the house itself, the roof and the walls, but also smaller parts like single tiles, bricks, windows or the door. In particular, we reviewed the zero-shot detectors DETR and OLN, including some extensions of the original DETR network.

DETR We explored a set-based object detector model called Detection Transformer (DETR) [2] which combines a CNN backbone with a transformer architecture. As seen in Figure 3 the DETR is a four-stage model consisting of a backbone, an encoder, a decoder and finally several prediction heads. In the first part of the model, the backbone, a CNN learns a 2D representation of the input image which is flattened and combined with a positional encoding. This is used as an input for the transformer encoder. Together with a number of learned positional embeddings, the output of the encoder is passed to the decoder. Finally, the decoder passes one embedding for each positional embeddings to a shared feed forward network (FFN). The FFN predicts bounding boxes with a class label or with the label "no object". DETR was trained using the COCO dataset. Object detectors that were trained on predefined categories tend to overfit to those and might have problems dealing with novel object categories that were not represented in the training data.

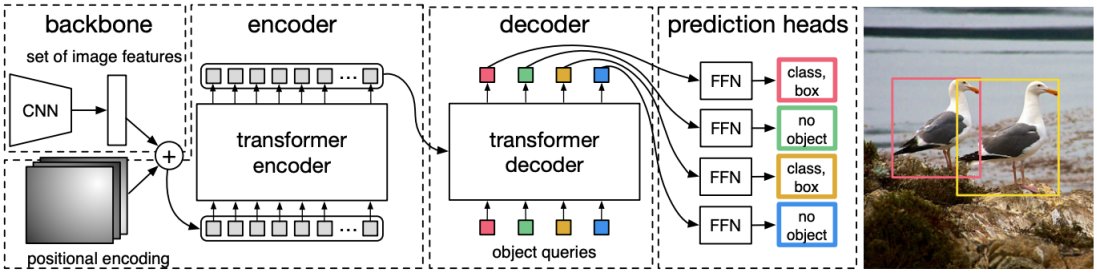


Figure 3: The DETR Model [2]

DETReg To reduce the dependence of the COCO dataset, we explored further extensions of DETR such as the class agnostic DETReg [30]. The central idea is that by generating priors similar to supervised object detection, DETReg improves object detection by transferring pretrained knowledge. However, DETReg could not achieve the same performance as DETR, and we realized that the dependence of COCO was not a real issue in our case. Since the bounding box predictions are processed in another cleaning and matching step where false bounding boxes can be filtered out, they do not need to be faultless in the first place. Also, we do not further use the predicted labels in our pipeline, which makes them irrelevant.

Deformable DETR It is also stated that DETR suffers from slow convergence in the training, which the Deformable DETR model [35] solves by introducing a so-called deformable attention module. After all, we decided against using this extension of DETR, since we used a pretrained object location network instead of training it from scratch.

OLN In search of another class agnostic approach, we reviewed the class agnostic Object Localization Network (OLN) [29]. Although the approach is rather simple, this model outperforms other state-of-the-art methods on unseen categories. The model is a 2-stage model consisting of a Feature Pyramid Network (FPN) [36] and a region-based stage. The structure itself is similar to a Faster R-CNN [12], where the classifiers of both stages are replaced with localization quality estimators. This adaption improves the generalization to novel objects from other datasets.

The feature maps extracted by FPN [36] are used to predict top generated region proposals, which are afterwards are fed into the regressor. We obtain IoU scores and bounding box coordinates at the output. In the end, we got very decent bounding boxes which could be used in the main pipeline, however based on the comparison with the DETR model 4, we decided to use the DETR bounding boxes instead.

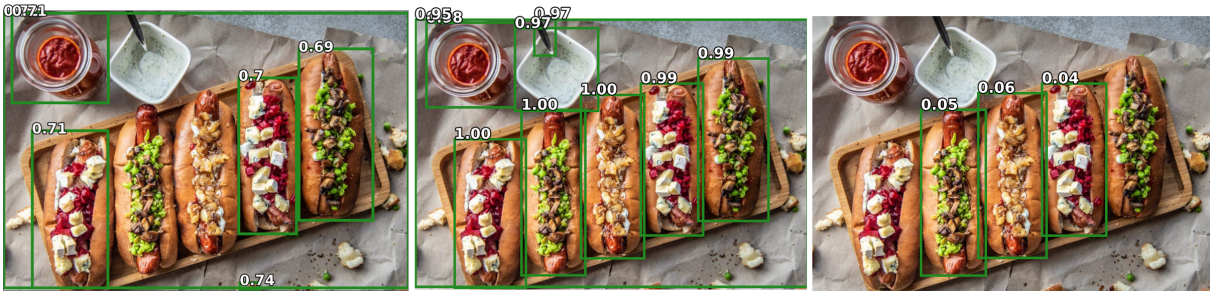


Figure 4: **Visualization of test results for hot dog category (not presented in COCO dataset) Test results from left to right: OLN, DETR, fine-tuned DETReg.**: DETR performs best, OLN can identify three out of five desirable objects with high IoU, and DETReg also provides reasonably good bounding boxes for novel category, but with low confidence.

3.1.2 One-Shot Detection

To get more relevant bounding boxes compared to the zero-shot approaches, we reviewed OS2D, a promising one-shot detector. In one-shot detection we are giving the model a query image of the object that it needs to detect. Doing so, the model should be able to predict better bounding boxes, since it is not searching in the dark, but instead knows what kind of object to detect.

OS2D [6], combines the dense grid of anchor locations of a Faster R-CNN [12] and SSD [14] object detectors. The key feature of this model is that detection and recognition are performed jointly without the need to define a general object. Instead, a good feature descriptor and transformation model are necessary.

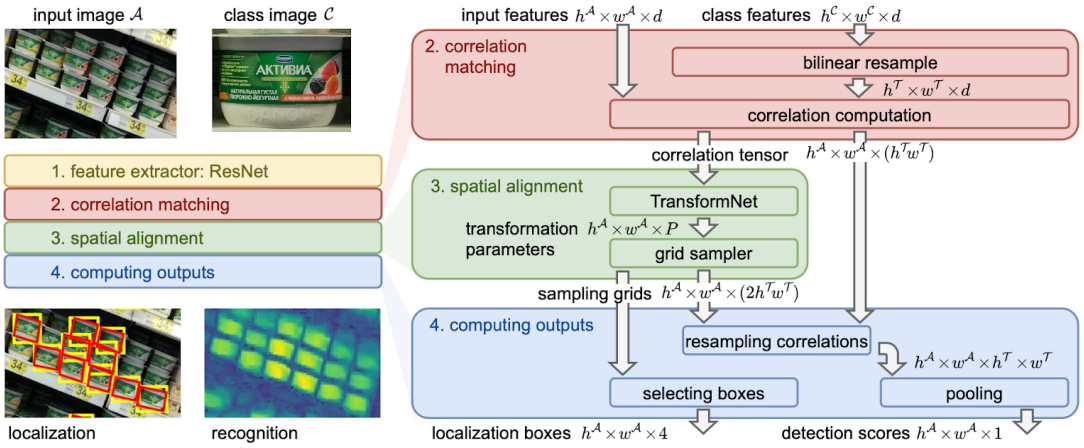


Figure 5: Inputs, outputs and main components of the OS2D model [6]

The components and the main architecture of the OS2D model are shown in figure 5. The yellow detection boxes at the bottom left correspond to the peaks in the score map, while the red parallelograms illustrate the corresponding affine transformations produced by the TransformNet. As shown in figure 5 the OS2D architecture consists of four steps: (1) extracting local features from both input and class images using a ResNet; (2) correlation matching of features; (3) spatially aligning features according to successful matches; (4) computing the localization bounding boxes and recognition score. [6]

3.1.3 Few-Shot Detection

Finally, our idea was to further improve the prediction of the bounding boxes by using multiple query images instead of only one. As explained in 2, there are multiple few-shot detection models. We decided to explore the RetinaNet [37] model with a ResNet-50 backbone, since it had an easy setup and is a well-cited object detection model.

RetinaNet RetinaNet is a one-stage detector, that is, a proposal-free network, that utilizes *focal loss* to highlight class imbalance during training. Focal loss is in turn calcu-

lated by applying a modulating term to the standard cross entropy, putting more focus on hard negative examples.

In our tests, the model demonstrated good results when using 10 query images but did not perform as well when only using three query images, as can be seen in Figure **??**. As the results of OS2D did not need as many images for a similar performance, we decided to not continue working with RetinaNet.

Even though we decided to proceed with other object localization frameworks, it would still be interesting to use a few-shot detector to perform experiments using a variable number of query images. When using a single query image there is no way to improve the level of generalization since the query becomes the visual descriptor to the model. Multiple query images could help the model to generalize better and make it less sensitive to the color and shape of objects. For example, the model could generalize and learn to detect the class "donuts" given multiple query images of donuts of different colors. In contrast, a one-shot detector might only be able to detect donuts with a specific frosting color, depending on the query image that was provided.

3.1.4 Meta learning

Finally, we explored a completely other idea for the region proposal generation, namely meta learning. Meta learning [38] is a sub-field of machine learning which embraces any type of learning based on prior knowledge from other tasks. There are different techniques to use meta learning, however, according to the no free lunch theorem [39], there is no universal approach. During our research, we looked into some of the most popular few-shot models using meta-learning, including Meta DETR [25], Meta R-CNN [26] and Context-Transformer [40].

Since all the corresponding repositories of these models did either miss pretrained checkpoints or code that was ready to use, we decided to implement our own model from scratch. The architecture of the model was inspired by the paper: "Few-shot object detection via feature reweighting" [21]. However, in contrast to the original model, we proposed an one-way class-agnostic localization detection network. In our setup, a query image and support images (one per class) were used as inputs. The bounding boxes of the query image corresponding to the objects were obtained from the support images. In a first step, the extracted meta-features from query images and support features were projected at the same space and multiplied channel-wise. This result was fed to the one-stage detector, which directly regressed the bounding box location for each class given the support images.

Due to time constraints, we could not finish training the model from scratch. Nevertheless, the current results can be seen in Appendix Figure 17. However, even though we did not end up including meta learning in our final experiments, this might be an interesting approach to improve the quality of the region proposals.

3.1.5 Included Object Localization Model

For the previously explained reasons, we decided to include DETR as a zero-shot detector and OS2D as an one-shot detector in our experiments.

DETR The DETR model in our experiments was trained on a subset of 80 COCO classes. It predicts bounding box candidates and the corresponding certainty scores given only the input image.

OS2D The OS2D network used in our experimental pipeline was trained on the Grozi-3.2k Dataset, a dataset of labeled products in Swiss supermarket shelves [41]. In addition to the input image, we feed a query image as an input, and receive the detected bounding boxes and certainty scores as an output. The samples in this dataset were all captured from a certain angle right across the shelves. Therefore, the objects to be detected are clearly visible and do not differ from one another.

In both cases the detected bounding boxes and certainty scores are propagated to the next module, the non-maximum suppression, in order to remove duplicates and increase the quality of the predicted bounding boxes.

3.2 Non-Maximum Supression

A very common technique used to reduce redundant bounding boxes around an object is Non-Maximum Supression (NMS). This problem arises because of how object detection is performed using the localization networks mentioned in the previous sections. They generate anchors of various shapes and sizes, although there is only one object existing in the scene. core aspect of NMS is to define a probability threshold which allows us to decide when to keep or discard a predicted bounding box. In the proposed pipeline, we use IoU score between each of the predicted bounding boxes and the ground truth and define a threshold for the same. Based on this, we discard bounding boxes that have a low score. Note that the lower the IoU score, the farther away the bounding box is from the ground truth.

Figure 6 shows a comparison bewtween all the bounding boxes predicted by DETR versus when the NMS algorithm is applied to the model output. In this case, the goal is to detect bananas object. We can make two major observations from Figure 6a:
1. The number of bounding boxes around the object is more than one,
2. Many of the predicted bounding boxes around the object are either around parts of the object or are around other objects in the given scene.

Figure 6b shows the same input image but with NMS applied to it. This helps us in optimizing our downstream pipeline by processing bounding boxes that strictly fall within the defined threshold.

(a) (b)

Figure 6: (a) DETR output vs (b) DETR output with NMS

3.3 Image Embeddings

The centerpiece of our pipeline is the image embedding matching module. To detect if a proposed bounding box shows the requested object or not, all bounding box candidates are compared to the query image using CLIP image embeddings [32]. Originally, CLIP uses an image and a text encoder to map image to text for zero-shot prediction. However, we eliminated the text input and only used the CLIP image encoder since we wanted to focus on working with image inputs for our zero-shot and few-shot experiments. The in-built CLIP image encoder takes an RGB image as an input and resizes it using bicubic interpolation and center crop. Finally, it generates an unique image embedding vector of shape $[768, 1]$.

We compare the image embeddings of the region proposals b_i with the image embedding q of our query image using the two most commonly used distance metrics, euclidean distance d_e 1 and cosine distance d_c 2.

$$d_e(b_i, q) = \sqrt{(b_i - q)^2} \tag{1}$$

$$d_c(b_i, q) = 1 - \frac{b_i \cdot q}{\|b_i\| \cdot \|q\|} \tag{2}$$

In the last step, we perform a matching using a variable threshold. The threshold ts calculates from the mean μ_d and standard deviation σ_d calculated over the computed distances between the query embedding q and all bounding box embeddings b_i. The sensitivity factor s indicates how many standard deviations are subtracted from the mean to determine the threshold level 3.

$$ts = \mu_d - s \cdot \sigma_d \tag{3}$$

If the distance $d(b_i, q)$ between the query image embedding q and the bounding box candidate embedding b_i does not exceed the computed threshold ts, they are predicted as a match. With increasing sensitivity factor s, the threshold decreases and the matching becomes more strict. We used several sensitivity factors in our experiments.

3.4 Captioning

The final stage of our proposed pipeline consists of an image captioning model that was added in order to retrieve keywords for the detected bounding boxes from input images. For this we used a pretrained ClipCap model [42] which generated a caption in natural language for each matched bounding box.

ClipCap is based upon CLIP [32] and proposes a lightweight captioning approach, making use of pretrained models for image and text processing. The key idea is to use the CLIP encoding as a prefix to the textual captions by employing a simple mapping network over the raw encoding, and then fine-tune a language model to generate valid captions. In addition, ClipCap utilizes a transformer architecture for the mapping network to avoid fine-tuning of the language model.

Due to time constraints, we could not fully explore how to utilize the outputs from the ClipCap model. Therefore, this was not part of our final experiments. However, the ideas related to the caption processing will be further discussed in Section 5.

4 Experiments and Results

For our experiments, we settled to focus on the two object localization networks DETR and OS2D, previously explained in Section 3.1. Besides comparing the impact of using different object localization networks, we tried to max out the model's performance by testing several parameter settings for the image embedding matching in our pipeline. Doing so, we examined the importance of the used distance metric and the sensitivity factor. We performed each experiment using subsets of two different object detection datasets, namely the *Oktoberfest Food Dataset* and the *Objects365 Dataset*, which we present in Section 4.1.

Since our pipeline requires a query image in the embedding matching step, we decided to use at least two query images per object class in order to minimize the impact of the query image choice. Furthermore, in experiments using the one-shot object detector OS2D we also need a query image to perform the detection itself. For simplicity, we made use of the same image samples as already collected for the matching. For each query image used in the one-shot object detection, we tested it with all available query images (usually two) in the matching stage, meaning we ran the model for a single sample at least four times.

In the end, we performed a final experiment using the best-performing parameters and a larger subset of approximately, 7000 images of the *Objects365 Dataset*.

4.1 Datasets

To compare our experiments and evaluate the performance of our pipeline, we used two highly challenging object detection datasets. The *Oktoberfest Food Dataset* and *Objects365 Dataset* differ significantly and are therefore suitable to examine strengths and weaknesses of our proposed pipeline.

4.1.1 Objects365

The *Objects365 Dataset* [5] is a large-scale object detection dataset consisting of over 600 000 training images and over 10 million labeled bounding boxes. It is one of the largest object detection benchmarks, containing 10 times as many bounding boxes as the famous COCO dataset [18]. In total, it consists of 365 object categories and eleven super categories, such as *foods* and *electronics*. In this project, we are only interested in the food category that contains 71 classes, ranging from fruit classes such as *apple* and *avocado* to more broad categories such as *dessert* or *pasta*. The *Objects365 Dataset* provides a wide variation of scenes and also a great variety of class objects within a class, which makes it a very challenging dataset 15.

4.1.2 Oktoberfest Food Dataset

The second dataset we used for our experiments is the *Oktoberfest Food Dataset* [43]. The data was recorded at a beer tent at the Oktoberfest and consists of 15 different object categories for food and drink items, such as *Bier* (Beer) and *Pommes* (French fries)16. The dataset contains over 2500 hand-annotated object annotations for 1100 images. Out of the 15 categories, none is identical to any of the COCO classes, but there are some classes that are similar. For example, *Curry-Wurst* is similar to the COCO *hot dog* class. However, it is noteworthy that the *Oktoberfest Food Dataset* varies from other object detection datasets, since all images provide a bird's-eye view.

4.2 Performance on Objects365 Subset Dataset

For our first experiments we used a small subset of the Objects365 Dataset. Focusing on 10 food classes, namely apple, bread, cake, cookies, donut, egg, french fries, hamburger, pizza, and sandwich, we handpicked 10 images to construct the test set. Since the difficulty level of the samples in the Objects365 Dataset is very high, we selected samples that are not too difficult but still challenging. For clarification, in the original dataset an apple could be hidden somewhere in the background of an image (see Figure 15 in Appendix), making it even difficult for the human eye to detect it. In addition to that, we chose samples where the target object looked similar each time to simplify the process of selecting query images. For instance, the constructed class "egg" only contained sunny-side-up eggs instead of samples with hardboiled eggs or egg shells.

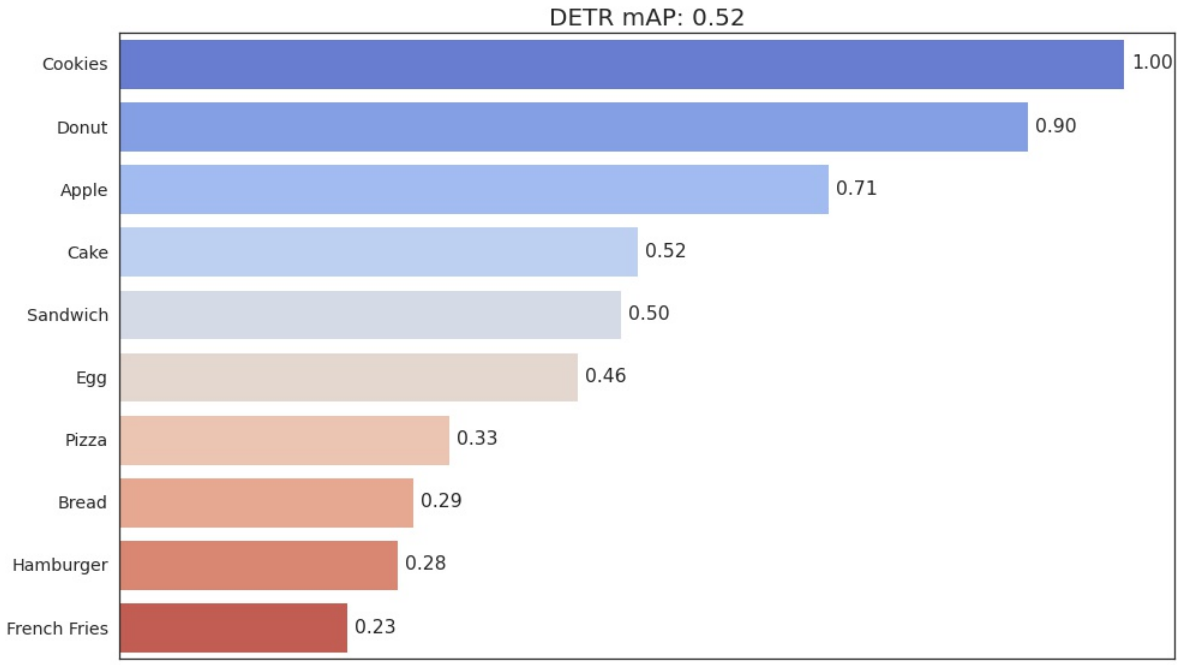


Figure 7: DETR model mAP values evaluated per class of Objects365 Subset Dataset

As seen in Figures 7 and 8 we observed that DETR performed slightly better on the objects365 subset, with an average mAP of 0.52 in comparison to the OS2D precision of 0.49. Both models perform very well on the *donut* class.

An interesting observation is that the OS2D model shows a drop in performance in comparison to the DETR models for multiple classes. This behavior is best to be seen for the category *apple*. The detection of the apples in itself is difficult for OS2D, probably due to the camera angle and the apples being ordered in tightly packed formations, in comparison to a single apple which was used as a query image for the region proposals which is. The DETR model seems to be able to both detect groups of apples as well single ones, probably because it was trained on COCO where the bounding boxes have demonstrated the possibility of this.

As can be seen in Figure 9, the ground truth boxes can either contain several apples or single ones. This demonstrates an issue of consistency in our testset. We cannot control how the bounding boxes are drawn, hence we cannot tell whether they are accurate as ground truth boxes for the provided query image. As a result, the mAP score might seem lower than it would be in reality.

4.3 Performance on Oktoberfest Food Dataset

The Oktoberfest Food Dataset, in contrast to the Objects365, is a smaller dataset with over 1000 images split into 15 classes, ranging from food classes such as *Burger* and *Pommes* to drink classes such as *Cola* (coke) and *Wasser* (water). Since this dataset

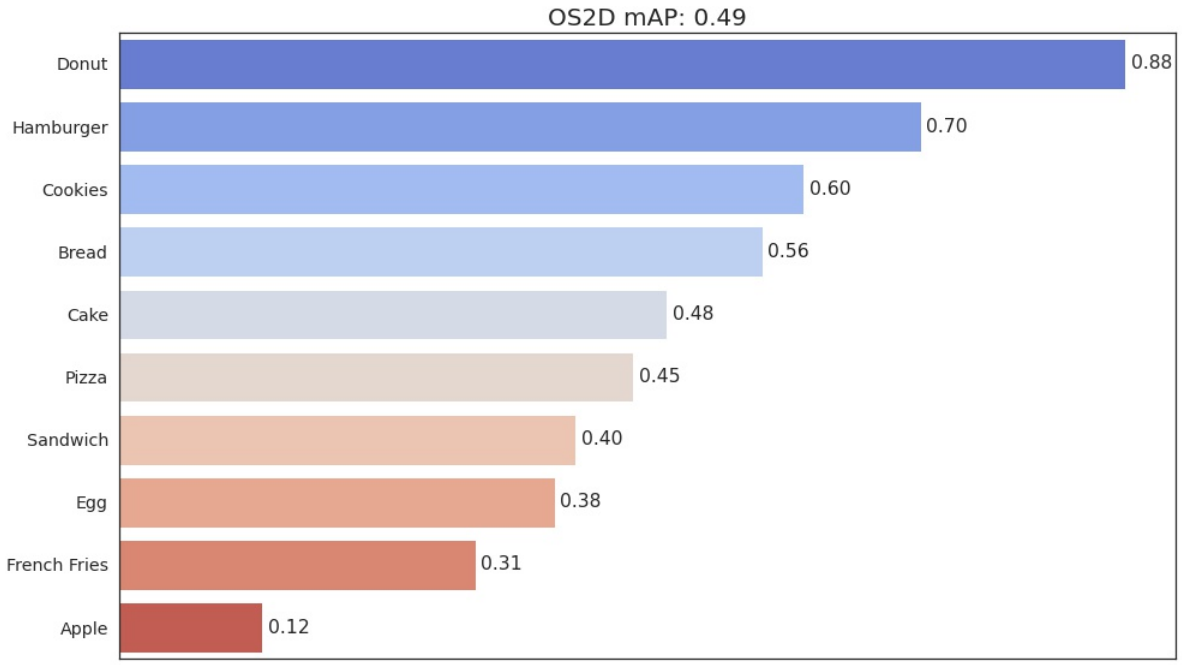


Figure 8: OS2D model mAP values evaluated per class of Objects365 Subset Dataset

contains only classes that are not part of any standard dataset such as ImageNet, COCO or Pascal-VOC, our assumption was that the performance of the zero-shot DETR model should be, on average, worse compared to the few-shot model OS2D.

Figure 9: Plot of failed detection of apples when running OS2D on the Objects365 Subset. The ground truth bounding boxes might either contain of several apples, or of single ones, making the prediction harder.

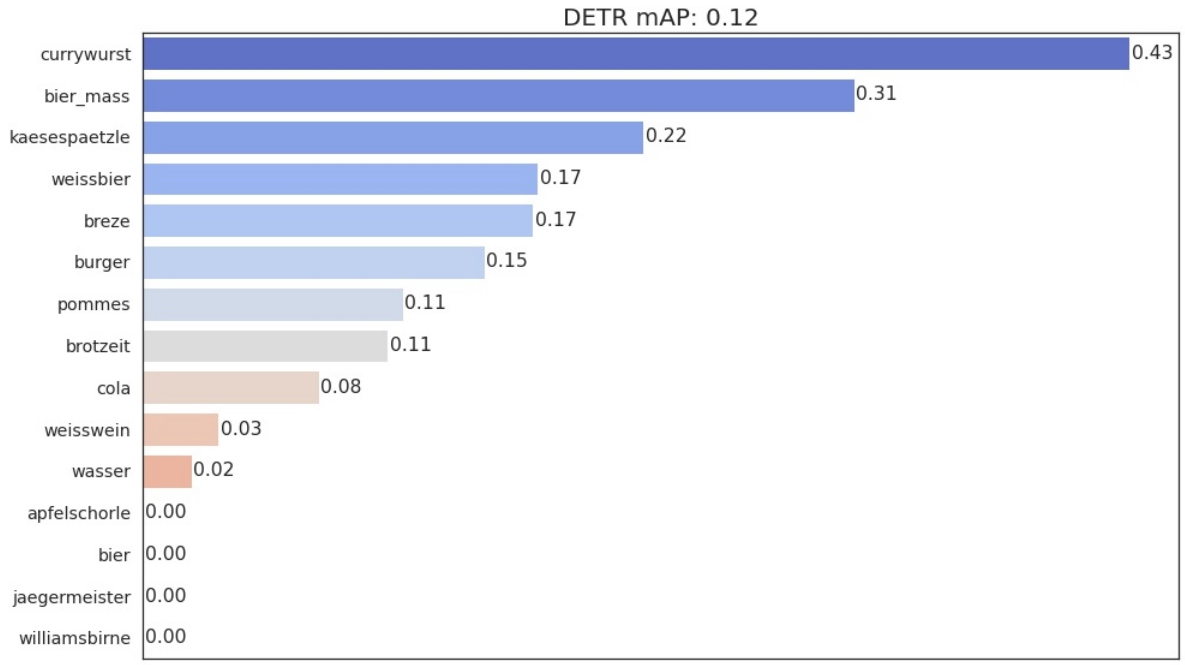


Figure 10: DETR model mAP values evaluated per class of Oktoberfest Food Dataset

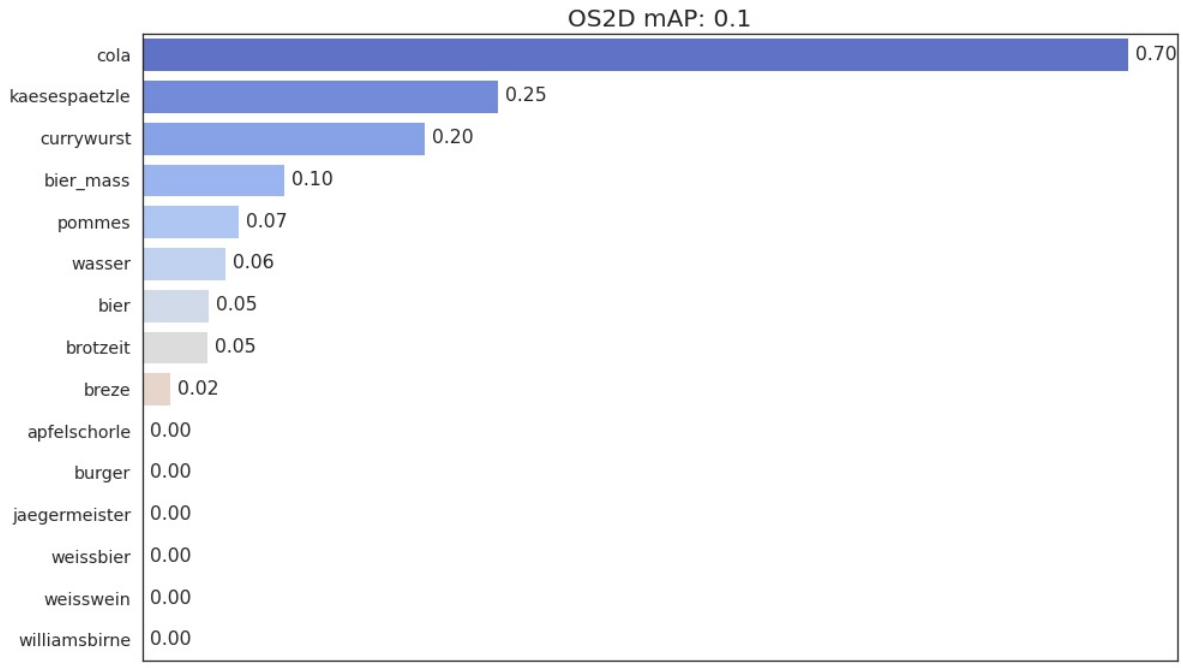


Figure 11: OS2D model mAP values evaluated per class of Oktoberfest Food Dataset

The preliminary results of our hypothesis can be seen in Figure 10 and Figure 11 where we can infer the following:

The average mAP score for DETR is lower compared to OS2D on the Oktoberfest Food Dataset. OS2D is a few shot detector. This means that for detecting objects in a given reference image, we provided the detector with two query images containing the object: one captured from the dataset itself and the second from the internet. Thus, the few-shot detector, through the query images already knows which feature descriptors are unique for each given object class. This is not the case for DETR, which is a zero-shot detector, thus explaining the lower score on novel classes.

The average mAP score for both DETR and OS2D are lower compared to the Objects365 Dataset. This can be explained by the quality of images, the size of both datasets and the difficulty of some classes. Especially the performance on the drink classes was worse than expected. A reason might be that it is a highly challenging task to predict bounding boxes for the drink classes. The classes *Williamsbirne* and *Wasser* are both clear liquids, served in a transparent glass which makes them often undetectable. Additionally, the dark green table and the lack of proper lightning considerably impede this task. It is worth noting that the detectors perform well on classes such as *Curry-wurst*. This might be related to the fact that it has feature resemblance to a hot dog which is a COCO class. However, since also the performance of a non-COCO class like *Kaesespaetzle* is remarkably good, detecting a dish served on a white plate might also be just a comparatively easier task for state-of-the-art detectors.

4.4 Final Experiment

To test our best-performing model, we used DETR as an object detection model, Euclidean distance, and a sensitivity factor of 0.5 for matching. Up to 100 images were picked from each subclass of the food category of the Objects365 Dataset, which increased the testset to a size of approximately, 7000 images. Since we randomly selected the images for this dataset, it contained more difficult samples than the hand-picked subset we used in our first experiments, which probably led to a lower performance.

Against the belief we had, the performance on classes that were also part of the COCO Dataset was not increased, although the object detector DETR was trained on a subset of COCO. As can be seen in Appendix Figure 21, the average mAP on COCO classes is 0.49, in comparison the average mAP on non-COCO classes is 0.37 (Appendix Figure 22). However, the mAP for the *donut* class has a considerably larger mAP of 0.86 than the rest of the COCO classes that have a mAP 0.38-0.5O, which be a reason for the higher mAP. In fact, despite being non-COCO, 11 classes outperform all COCO classes.

4.5 Comparison between DETR and OS2D

As mentioned in Section 3.1.5, the OS2D model was trained on the Grozi 3.2k Dataset, while DETR was trained on a subset of the COCO Dataset. As seen in Table 1 the pipeline

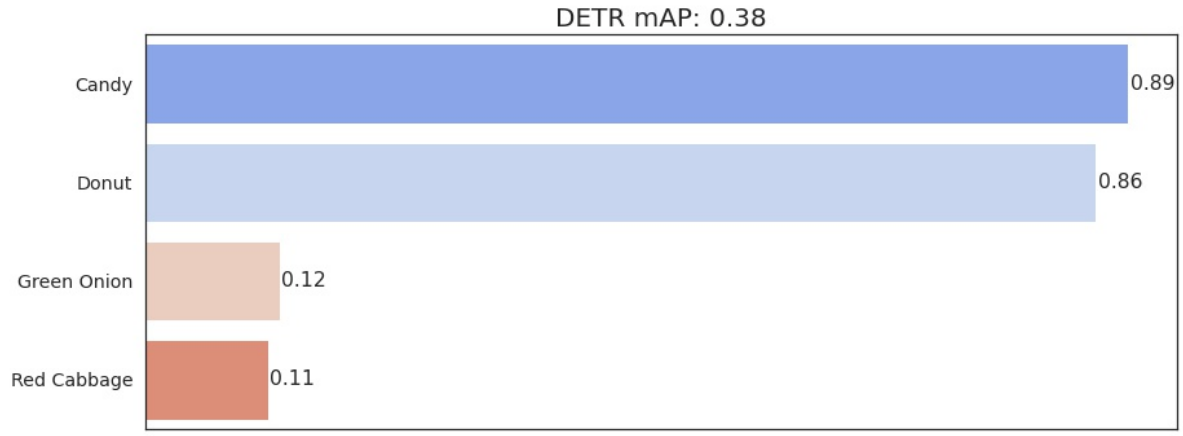


Figure 12: A comparison of the mAP values of the DETR model on Objects365 Dataset of the best and worst performing classes.

Figure 13: A comparison of the mAP values on Objects365 Dataset of the best and worst performing COCO classes.

using DETR as an object detector surpasses OS2D in all metrics by a large margin. While the DETR achieves an AP of 0.78 in its best configuration, OS2D only reaches an AP of 0.49. This behaviour was expectable, since the COCO Dataset resembles the Objects365 Dataset but not the Grozi 3.2k Dataset.

In contrast, the pipeline using OS2D outperforms the one using DETR on the Oktoberfest Food Dataset, as Table 2 indicates. One possible explanation is that OS2D creates better bounding boxes which affect the performance of the whole model. Since OS2D is a one-shot detection model, alongside to the input image it receives a query image, which acts as a guide for the model in terms of what to detect. In addition, the samples in the Oktoberfest Food Dataset are similar to the samples from the dataset OS2D was trained on. While the images in the Oktoberfest Food Dataset are all taken from a birds-eye-view, the once in the Grozi 3.2k Dataset are taken completely from the front which leads to a very similar image composition since all objects are clearly visible the specific angle.

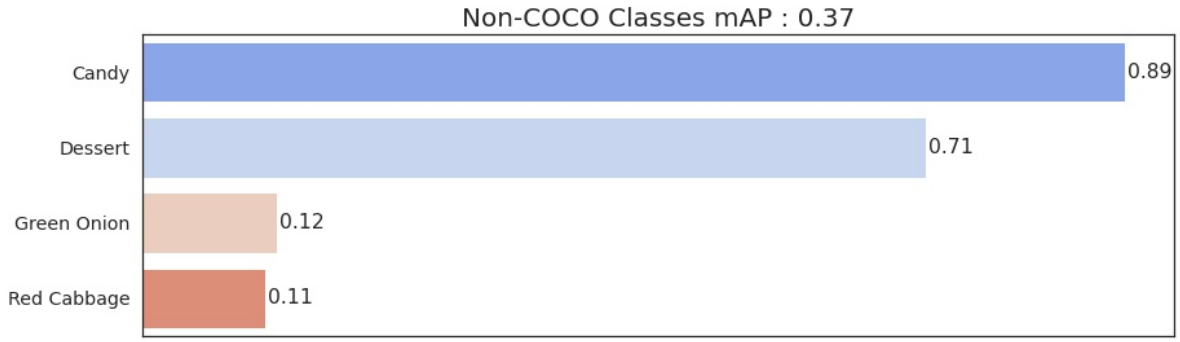


Figure 14: A comparison of the mAP values on Objects365 Dataset of the best and worst performing Non-COCO classes.

OLN	metric	sensitivity factor	AP	AR	F1
DETR	cosine	0	0.70	0.26	0.38
DETR	cosine	0.5	0.71	0.26	0.38
DETR	cosine	1	0.75	0.24	0.36
DETR	cosine	1.5	0.78	0.21	0.33
DETR	euclidean	0	0.44	0.43	0.44
DETR	euclidean	0.5	0.52	0.39	0.45
DETR	euclidean	1	0.67	0.33	0.44
DETR	euclidean	1.5	0.76	0.26	0.39
OS2D	cosine	0	0.44	0.07	0.13
OS2D	cosine	0.5	0.41	0.06	0.11
OS2D	cosine	1	0.4	0.04	0.07
OS2D	cosine	1.5	0.23	0.02	0.04
OS2D	euclidean	0	0.5	0.09	0.16
OS2D	euclidean	0.5	0.46	0.08	0.14
OS2D	euclidean	1	0.47	0.06	0.11
OS2D	euclidean	1.5	0.32	0.03	0.06

Table 1: Performance metrics comparison of Objects365 Subset Dataset

4.6 General Observations

In summary, we could proof that the pipeline we came up with and the approach of using a matching of CLIP image embeddings works for zero-shot and one-shot detection. However, to achieve a better performance and more robust results the pipeline needs more fine-tuning. As the foregoing discussions showed, the choice of the object detector and the query image are both crucial.

OLN	metric	sensitivity factor	AP	AR	F1
DETR	cosine	0	0.16	0.02	0.04
DETR	cosine	0.5	0.16	0.02	0.04
DETR	cosine	1	0.18	0.02	0.04
DETR	cosine	1.5	0.17	0.02	0.04
DETR	euclidean	0	0.06	0.06	0.06
DETR	euclidean	0.5	0.09	0.06	0.07
DETR	euclidean	1	0.12	0.05	0.07
DETR	euclidean	1.5	0.13	0.05	0.07
OS2D	cosine	0	0.14	0.04	0.06
OS2D	cosine	0.5	0.15	0.04	0.06
OS2D	cosine	1	0.17	0.04	0.07
OS2D	cosine	1.5	0.24	0.04	0.06
OS2D	euclidean	0	0.1	0.08	0.09
OS2D	euclidean	0.5	0.13	0.08	0.10
OS2D	euclidean	1	0.17	0.08	0.11
OS2D	euclidean	1.5	0.23	0.07	0.11

Table 2: Performance metrics comparison of Oktoberfest Food Dataset

5 Future Work

5.1 Processing ClipCap Captions

As we stated in Section 3, we could not explore how to utilize the outputs from the ClipCap model since we did not set metrics to measure its performance. This means that even tough we set up the model to work in our pipeline, we did not generate captions for the samples during our experiments. In this section, our ideas about how to integrate ClipCap into the pipeline and evaluate meaningful results will be discussed.

First of all, ClipCap produces full sentences as captions, and only a few of the words in the captions are relevant to our task. For example, in Figure 23, the words of interest for our objects of pink donuts are: "pink", "donut", "frosting" and "sprinkles". We don't need the other words in the sentence. Therefore, we need a system that extracts the words of interest from the caption sentences. For this task, we can first use a tokenizer to extract all the tokens from the sentences. Then, we get word embeddings from a model such as word2vec and also generate image embeddings for the object that the caption belongs to with CLIP. After this, we can train a model with the caption's word embeddings and object's image embeddings as input and the embeddings of the words of interest as the target output. Of course, this idea requires us to have labeled data in the described format, therefore could not be achieved in the given time frame.

5.2 Meta Learning

Since there was a time constraint, the meta-learning algorithm presented in the work could not be sufficiently trained. The core idea is to revisit the architecture of the existing outline. Specifically, the existing architecture allow identifying only one bounding box per one class of support image, but could be generalized to recognize multiple bounding boxes. Additionally, we would run the training process in the cluster environment. By training on different datasets, we can induce different priors, trying other architectures, which might be based on Meta-RCNN, [26] and Meta-DETR[25]. Choosing different pretrained backbone models, other than those offered in the work, can also bring about more visible results.

In the end, Meta-Learning algorithms can be integrated into the final model to induce class agnostic properties and facilitate generalization. Such models also have nice interpretability and can be used to further explain the overall results produced by the pipeline.

5.3 Other Object Localization Networks

As stated before, the performance of our pipeline highly depends on the choice of the object localization network. To achieve better results, a first step would be to improve the prediction of the bounding boxes in the first step of the pipeline.

One interesting approach would be to extend and improve the models we have already taken into account. The zero-shot detection models OLN and DetReg, mentioned in 3.1.1, might only need some fine-tuning to produce better results. According to the previous discussion, DetReg's architecture is based on DETR. It would appear that if sufficient training time and proper hyperparameter tuning is given, it should be possible to outperform the embedded DETR and therefore provide more meaningful bounding box predictions.

Furthermore, there are models and approaches which we could not include in our research, but are still worth mentioning, including anchor-free object detection models, such as Center-Net[44], Corner-Net[45], CentripetalNet[46].

6 Conclusion

The aim of this project was to come up with an universal object detection architecture that could detect any novel object in an image, being provided not more than one reference sample. In contrast to other state-of-the-art methods, we presented an approach whose unique feature is the utilization of CLIP image embeddings for a matching between the reference sample and all detected objects in the input image.

There are three modules in our pipeline: an object localization network to detect all objects in an input image, a matching component to compare CLIP image embeddings of

all detected objects with the reference image, and finally a captioning module to create meaningful textual descriptions of the objects. During the process of assembling the pipeline, we reviewed and evaluated multiple state-of-the-art object localization networks and decided to use for our experiments zero-shot DETR model, and an one-shot detector OS2D model.

Besides comparing the impact of using those object localization networks, we tried to max out the model's performance by testing several parameter settings in our pipeline for the image embedding matching. To evaluate the performance, we used two highly challenging and different object detection datasets, the Oktoberfest Food Dataset and the Objects365 Dataset. Each of them introduced its own challenges, which enabled us to fully examine the strengths and weaknesses of our approach.

In summary, we could prove that the pipeline we came up with and the idea of using a matching of CLIP image embeddings works as a one-shot detector. However, as expected, the pipeline needs more fine-tuning to achieve better performance and more robust results. As the foregoing discussions showed, the choice of the object detector and the query image are both crucial in our proposed set-up and must be considered carefully.

Finally, to improve our pipeline, more research must be done, especially on the object localization network. On this matter, It might be worth investigating completely different approaches, such as Meta-learning.

References

[1] "Precitaste inventory management system," 2021. [Online]. Available: https://www.youtube.com/watch?v=We_dFQhNDew

[2] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[4] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. DollÃ¡r, "Microsoft coco: Common objects in context," 2014. [Online]. Available: https://arxiv.org/abs/1405.0312

[5] S. Shao, Z. Li, T. Zhang, C. Peng, G. Yu, X. Zhang, J. Li, and J. Sun, "Objects365: A large-scale, high-quality dataset for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 8430–8439.

[6] A. Osokin, D. Sumin, and V. Lomakin, "Os2d: One-stage one-shot object detection by matching anchor features," in *European Conference on Computer Vision*. Springer, 2020, pp. 635–652.

[7] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–I.

[8] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. Ieee, 2005, pp. 886–893.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[10] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[11] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[12] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[14] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[15] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4203–4212.

[16] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee, "A survey of modern deep learning based object detection models," *Digital Signal Processing*, p. 103514, 2022.

[17] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *International journal of computer vision*, vol. 128, no. 2, pp. 261–318, 2020.

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

[19] A. Gupta, P. Dollar, and R. Girshick, "LVIS: A dataset for large vocabulary instance segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[20] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," no. 0. Toronto, Ontario: Technical report, University of Toronto, 2009.

[21] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell, "Few-shot object detection via feature reweighting," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 8420–8429.

[22] X. Wang, T. E. Huang, T. Darrell, J. E. Gonzalez, and F. Yu, "Frustratingly simple few-shot object detection," *arXiv preprint arXiv:2003.06957*, 2020.

[23] B. Sun, B. Li, S. Cai, Y. Yuan, and C. Zhang, "Fsce: Few-shot object detection via contrastive proposal encoding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7352–7362.

[24] L. Qiao, Y. Zhao, Z. Li, X. Qiu, J. Wu, and C. Zhang, "Defrcn: Decoupled faster r-cnn for few-shot object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 8681–8690.

[25] G. Zhang, Z. Luo, K. Cui, and S. Lu, "Meta-detr: Image-level few-shot object detection with inter-class correlation exploitation," *arXiv preprint arXiv:2103.11731*, 2021.

[26] X. Yan, Z. Chen, A. Xu, X. Wang, X. Liang, and L. Lin, "Meta r-cnn: Towards general solver for instance-level low-shot learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9577–9586.

[27] C. Tan, X. Xu, and F. Shen, "A survey of zero shot detection: methods and applications," *Cognitive Robotics*, vol. 1, pp. 159–167, 2021.

[28] Y. Xian, B. Schiele, and Z. Akata, "Zero-shot learning-the good, the bad and the ugly," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4582–4591.

[29] D. Kim, T.-Y. Lin, A. Angelova, I. S. Kweon, and W. Kuo, "Learning open-world object proposals without learning to classify," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5453–5460, 2022.

[30] A. Bar, X. Wang, V. Kantorov, C. J. Reed, R. Herzig, G. Chechik, A. Rohrbach, T. Darrell, and A. Globerson, "Detreg: Unsupervised pretraining with region priors for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 605–14 615.

[31] X. Gu, T.-Y. Lin, W. Kuo, and Y. Cui, "Open-vocabulary object detection via vision and language knowledge distillation," *arXiv preprint arXiv:2104.13921*, 2021.

[32] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8748–8763.

[33] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig, "Scaling up visual and vision-language representation learning with noisy text supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 4904–4916.

[34] S. Esmaeilpour, B. Liu, E. Robertson, and L. Shu, "Zero-shot out-of-distribution detection based on the pretrained model clip," in *Proceedings of the AAAI conference on artificial intelligence*, 2022.

[35] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020.

[36] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[37] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[38] J. Vanschoren, "Meta-learning," in *Automated machine learning*. Springer, Cham, 2019, pp. 35–61.

[39] C. Giraud-Carrier and F. Provost, "Toward a justification of meta-learning: Is the no free lunch theorem a show-stopper," in *Proceedings of the ICML-2005 Workshop on Meta-learning*, 2005, pp. 12–19.

[40] Z. Yang, Y. Wang, X. Chen, J. Liu, and Y. Qiao, "Context-transformer: tackling object confusion for few-shot detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 12 653–12 660.

[41] M. George and C. Floerkemeier, "Recognizing products: A per-exemplar multi-label image classification approach," in *European Conference on Computer Vision*. Springer, 2014, pp. 440–455.

[42] R. Mokady, A. Hertz, and A. H. Bermano, "Clipcap: Clip prefix for image captioning," *arXiv preprint arXiv:2111.09734*, 2021.

[43] A. Ziller, J. Hansjakob, V. Rusinov, D. Zügner, P. Vogel, and S. Günnemann, "Oktoberfest food dataset," *arXiv preprint arXiv:1912.05007*, 2019.

[44] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6569–6578.

[45] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 734–750.

[46] Z. Dong, G. Li, Y. Liao, F. Wang, P. Ren, and C. Qian, "Centripetalnet: Pursuing high-quality keypoint pairs for object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 519–10 528.

A Appendix

A.1 Performance Metrics

The *Intersection over Union* (IoU) is a widely used evaluation metric in object detection problems and measures how much the predicted bounding box overlaps with the ground truth. An IoU of 1 reports that the prediction and ground truth overlap completely, while an IoU of 0 shows that prediction and ground truth do not overlap at all. If the IoU exceeds a certain IoU threshold the prediction is counted as positive match, otherwise it is counted as negative.

Average Precision (AP) and *Average Recall* (AR) correspond the average precision and recall for IoU thresholds from 0.5 to 0.95 with a step size of 0.05 calculated for each class. The *Mean Average Precision* (mAP) is the averaged AP over all classes.

A.2 Additional Figures

Figure 15: Examples of *apple* category from Object365 Dataset

Figure 16: Examples from Oktoberfest Food Dataset

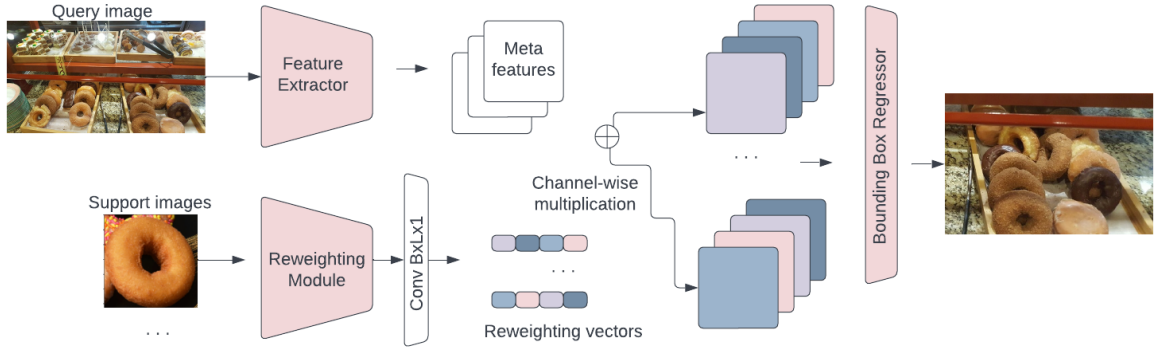


Figure 17: Architecture and test results of our class-agnostic one-shot meta-learning model. As we can see, the model is not yet able to detect individual objects, but it localizes the location of the donut instead of a random location.

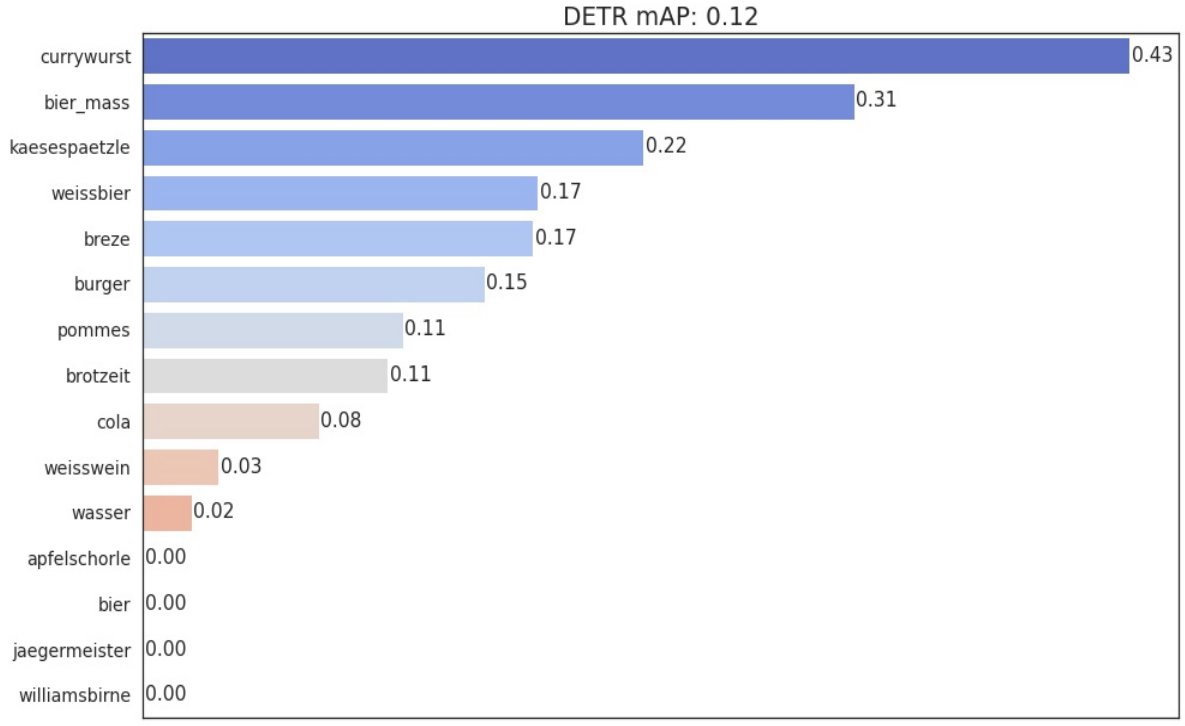


Figure 18: DETR model mAP values evaluated per class of Oktoberfest Dataset

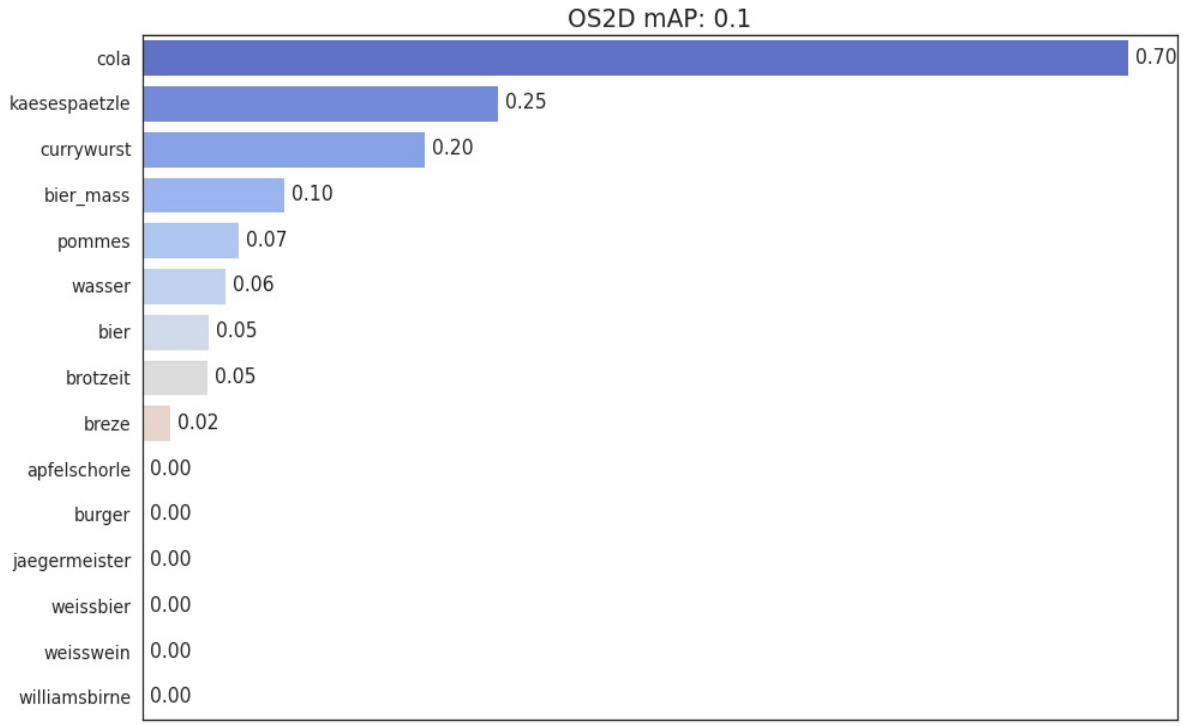


Figure 19: OS2D model mAP values evaluated per class of Oktoberfest Dataset

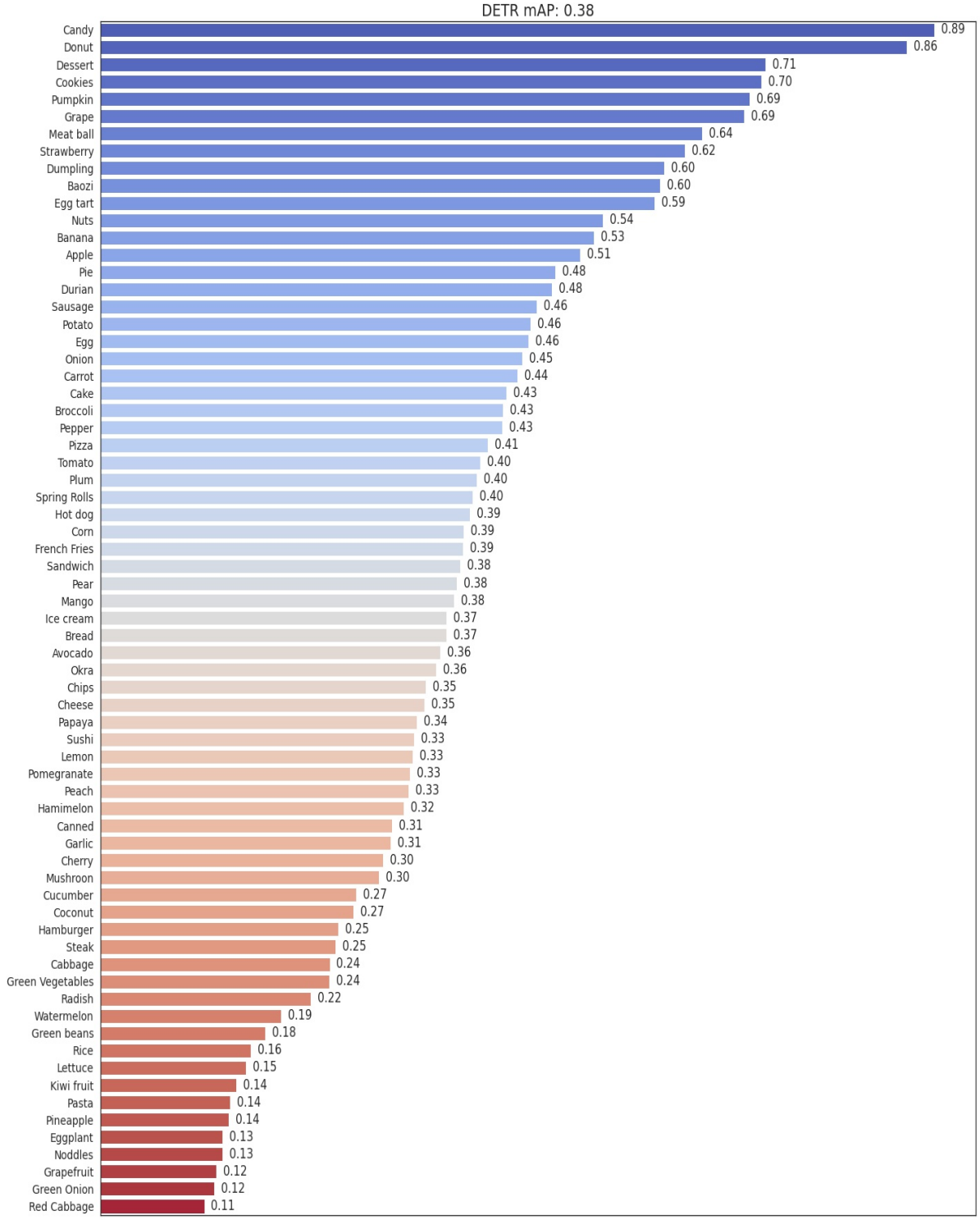


Figure 20: DETR model mAP values evaluated per class of Objects365 Dataset

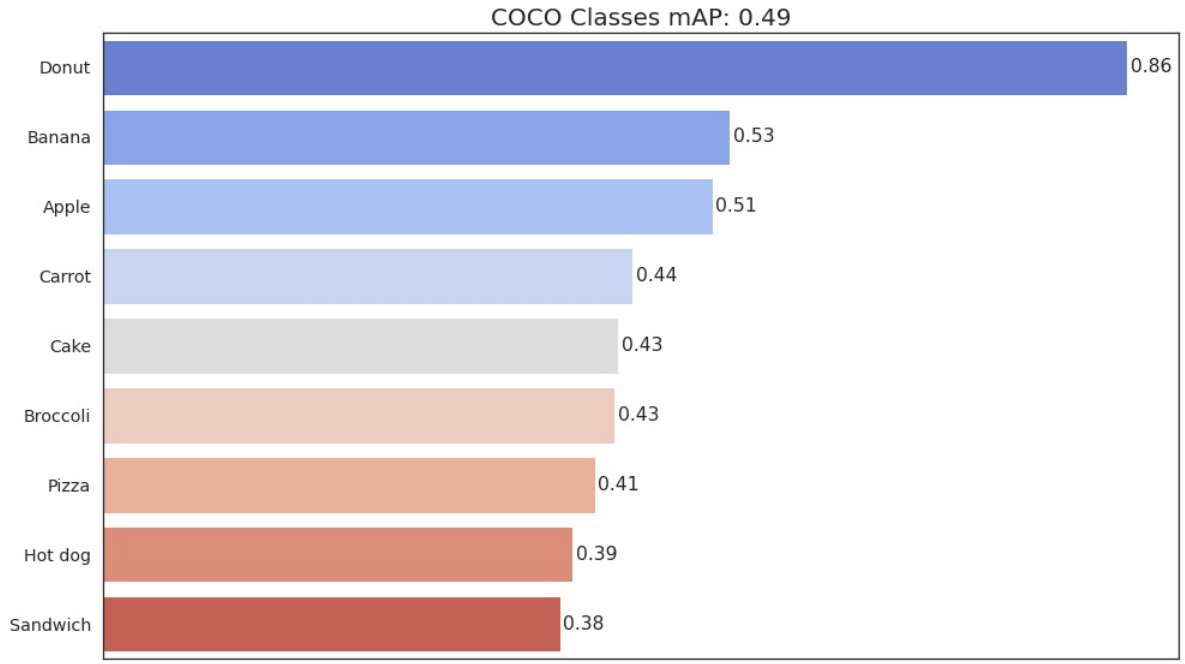


Figure 21: DETR model mAP values of COCO classes, evaluated per class of Objects365 Dataset

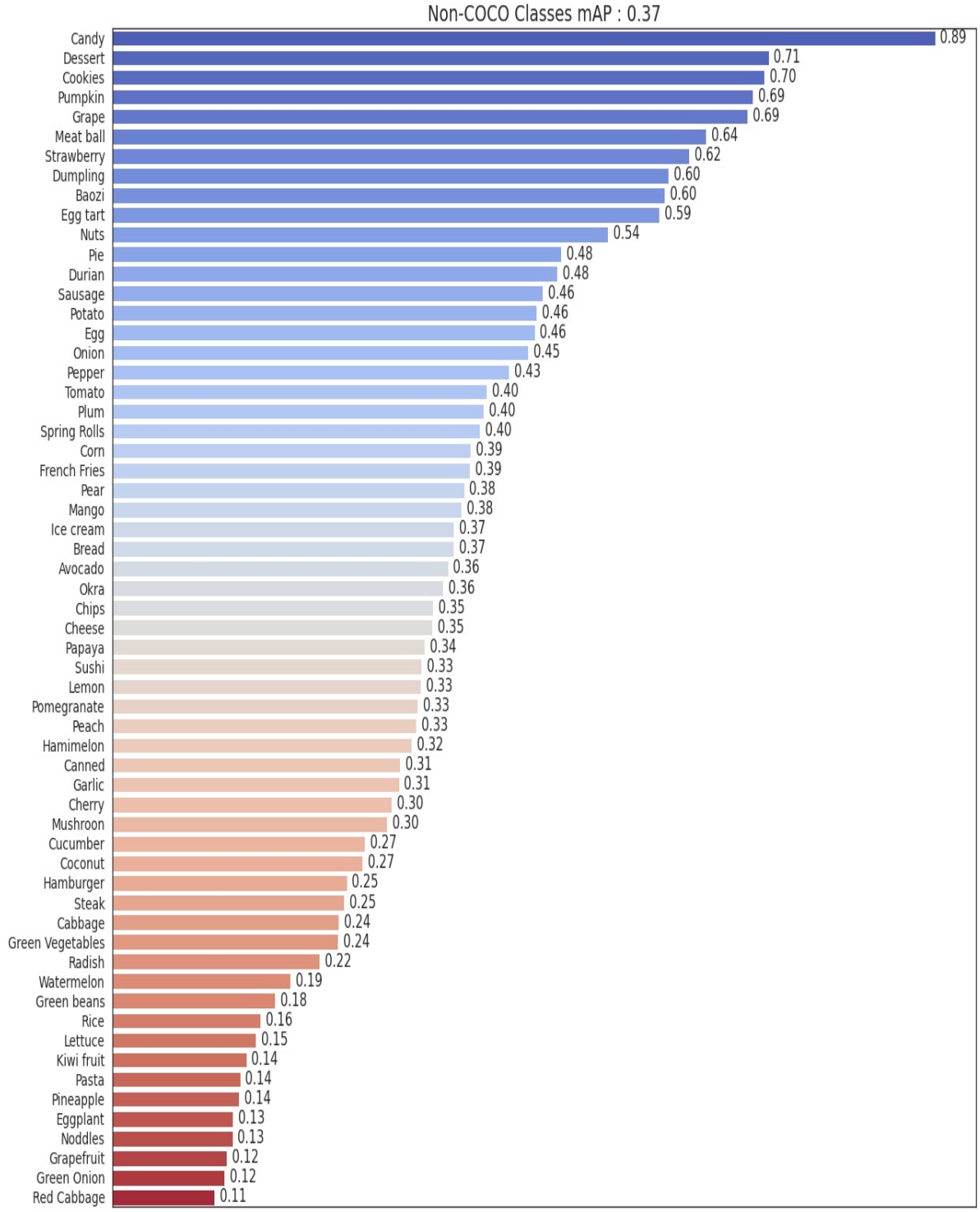


Figure 22: DETR model mAP values of non-COCO classes, evaluated per class of Objects365 Dataset


```
0: A pink plate topped with lots of donuts covered in frosting.
1: A donut with sprinkles and a bite taken out.
```

Figure 23: Example captions of two donuts generated by ClipCap