University of Tartu

Faculty of Science and Technology

Institute of Technology

Glib Manaiev

Improvement of small objects detection

Bachelor's Thesis (12 ECTS)

Curriculum Science and Technology

Supervisor:

Assoc. Prof., PhD Cagri Ozcinar

Tartu 2022

Abstract

Improvement of small object detection

Over the last decade, there has been done a lot of work on the improvement of neural networks that are working on object detection. Although detection performance has increased overall, small objects are still detected considerably less accurately than large ones. This problem is relatively understudied, considering how crucial is the detection of small objects in applications like self-driving cars. This thesis aims to study and compare the efficiency of three different approaches and their combinations in making lightweight model with increased small object detection performance. Used methods include modification of the training dataset by oversampling small objects using copy-paste data augmentation and altering the training procedure using channel-wise knowledge distillation and weighted loss. Results show that each method was able to increase the performance of small object detection when applied individually, but the best relative improvement in the performance of 29% was achieved by using channel-wise knowledge distillation combined with copy-paste data augmentation, which has not previously been considered in the literature.

CERCS:

T111 Imaging, image processing

Keywords:

Machine learning, Object detection, Data augmentation, Knowledge distillation, Weighted Loss

Kokkuvõte

Väikeste objektide tuvastamise paranemine

Viimasel kümnendil on tehtud palju tööd, et täiustada tehisnärvivõrke, mis käsitlevad objekti tuvastamist. Kuigi tuvastamise jõudlus on üldiselt suurenenud, siis väikseid objekte tuvastatakse siiani oluliselt ebatäpsemalt kui suuri. Seda probleemi on küllaltki vähe uuritud, arvestades kui tähtis on väikeste objektide tuvastamine kasutusvaldkondades nagu isesõitvad autod. Käesolev töö uurib ja võrdleb kolme erineva lähenemisviisi ja nende omavaheliste kombinatsioonide efektiivsust väikse objekti tuvastamise parandamisel. Kasutatud meetodid hõlmavad treenimisandmestiku modifitseerimist väikeste objektide ülediskreetimise kaudu, kasutades andmete suurendamist kopeerimise ja kleepimise teel ning treenimisprotseuuri muutmist, kasutades kanalipõhist teadmisülekannet ja kaalutud kadu. Iga meetod tõstis väikeste objektide tuvastamise jõudlust kui neid kasutati individuaalselt, kuid parim suhteline jõudluse suurenemine 29% võrra saavutati, kui kasutati kanalipõhist teadmisülekannet kombineerituna andmete suurendamisega kopeerimis-kleepimis-meetodil. Viimast pole varasemalt kirjanduses käsitletud.

CERCS:

T111 Kuvameetodid, pilditöötlus

Märksõnad:

Masinõpe, Objektituvastus, Andmesuurendus, Teadmisülekanne, Kaalutud kadu

Contents

Abbreviations

AP – average precision

AR – average recall

CNN – convolutional neural network

CPDA – copy-paste data augmentation

FN – false negative

FP – false positive

IoU – intersection over union

KD – knowledge distillation

KL divergence - Kullback–Leibler divergence

mAP – mean average precision

mAR – mean average recall

TP – true positive

Introduction

Small object detection plays an important role in applications such as aerial and satellite imagery analysis, especially in autonomous vehicles where the safety of pedestrians and car passengers depends on it.

Recently developed novel approaches and architectures made networks faster and more accurate [1], but there is still a noticeable gap in quality between the detection of small and large objects. In Table 1 you can see the most commonly used division of objects by the area of the rectangle they can be bounded with. You can see the top submissions for the Microsoft COCO object detection challenge [2] in Table 2, for small objects average precision (AP) and average recall (AR) are almost two times less than for large ones.

Table 1: The definitions of the small, medium and large objects in MS COCO.

	Min area	Max area
Small object	0	32 × 32
Medium object	32 × 32	96 × 96
Large object	96 × 96	∞

Table 2: Difference in average precision and recall for small and large objects in top-submissions of MS COCO object detection challenge.

Place	AP	AP_{small}	AP_{medium}	AP_{large}	AR	AR_{small}	AR_{medium}	AR_{large}
1	0.588	0.407	0.616	0.720	0.747	0.591	0.780	0.875
2	0.578	0.399	0.605	0.706	0.736	0.577	0.768	0.861
3	0.553	0.378	0.583	0.668	0.724	0.555	0.755	0.859
4	0.550	0.377	0.578	0.669	0.730	0.565	0.761	0.860
5	0.547	0.362	0.576	0.685	0.733	0.552	0.768	0.878

The reasons for the relatively poor performance of small objects detection are imperfections in architectures of neural networks, constraints in sizes of neural networks caused by limitations in computational resources and features of the datasets, on which networks are trained, such as

a low proportion of small objects on images and bad image quality [3]. There are many available approaches designed to tackle these problems, but most of them were not used to improve the detection of small objects in particular.

This thesis aims to study and compare the efficiency of different approaches and their combinations in improving the performance of small object detection by existing convolutional neural networks, without impairing their speed to make them suitable for real-time applications like self-driving cars. Three different approaches were studied:

1. Small objects oversampling using copy-paste training data augmentation. This method is addressing the shortcomings of datasets, taking an image containing small objects and copy-pasting each small object one time, saving new image along with the original one, thus multiplying both number of images with small objects on them and small objects. The approach was able to increase the performance of small objects detection by 7%.
2. Channel-wise knowledge distillation. The idea of this method is to transfer the knowledge learned by the larger 'teacher' network to the smaller 'student' network. It achieved 23% improvement on object detection of small objects.
3. Weighted loss. It tries to make network focus more on learning to detect small objects, stealing its attention from medium and large objects, by increasing training loss for small objects. This method increased the performance of small object detection by 5%.

However, even better results were achieved when approaches were used in combination. The biggest rise in the performance of small objects detection of almost 29% was achieved using channel-wise knowledge distillation together with copy-paste data augmentation.

The thesis is structured into five main parts. Chapter 1 gives an overview of the literature needed to deeper understand the problem and the methods used. Chapter 2 describes dataset and networks that were used in the experiments, as well as the experiments themselves. The experimental results and their discussion are presented in Chapter 3. Chapter 4 concludes the study and discusses future work.

1 Literature review

1.1 Object detection

Currently, object detection, instance segmentation and several similar tasks are among of the most important and widely used applications of machine learning algorithms. They are a key part of many tasks that rely on computer vision, such as medical imaging [4], self-driving cars [5], satellite image analysis [6] and many more. There are many machine learning algorithms that were developed for solving this task and the decision to use one algorithm or another depends greatly on the specific problem as well as on available computational resources. However, nowadays, neural networks (NN) and convolutional neural networks (CNN) in particular are becoming a universal solution for these kinds of problems [7]. Not so long ago, neural networks were very demanding on computing resources, and besides, the imperfection of the architecture did not allow them to effectively cope with the tasks set. But in the last decade, a large amount of research has been done on the design and optimization of the architectures of neural networks, which has significantly improved their performance in terms of both speed and accuracy [8]. In addition, advances in hardware have made it possible to apply them in real-time.

Object detection is a machine learning task for determining the categories and locations of the objects present in an image or video. It consists of two subtasks object localization and their classification [9]. The task of object localization is to find and bound all the areas of the frame that are most likely to contain objects. The output of object localization is a vector of type (x, y, w, h) that represents the so-called "bounding box", where x, y are coordinates of the center of the region of interest and w, h are the width and height of the rectangle that bounds this region, as shown on Figure 1. The task of object classification is to assign an object class to every region that was detected during the localization step as well as confidence in this decision. The prediction confidence values are subsequently used to threshold the resulting predictions to remove those that the neural network is least confident about.

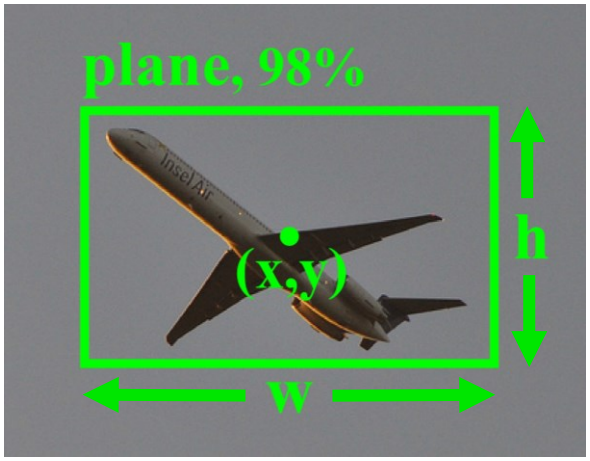
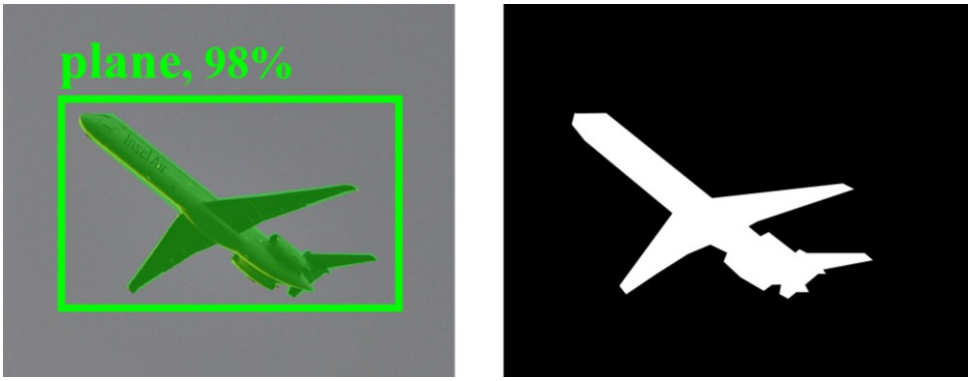


Figure 1. Object detection visualization. Outputs of object detection are bounding box, object class and confidence in prediction. Image is taken from MS COCO dataset.

An extension of the object classification problem is object segmentation [10], often referred to as instance segmentation. The difference from an object segmentation is that during the object localization step, instead of or in addition to bounding the region that contains an object with a rectangle, each individual pixel within this region is either assigned to the object or not, creating the so-called instance mask, as shown on Figure 2(a). Instance mask can be outputted in the form of a binary mask that is a matrix of the size of the original image that consists of ones and zeros, where the ones represent pixels that belong to the object of interest, as shown on Figure 2(b). The other way of returning an instance mask is the polygon format, which is an ordered list of x and y coordinates of vertices of a polygon that approximates the shape of the instance's mask outline.



(a) Prediction visualization (b) Binary mask

Figure 2. Object segmentation. Outputs of object segmentation are mask, object class and confidence in prediction. Bounding box coordinates can be calculated from the mask.

1.2 Detector types

1.2.1 Two-stage detectors

In general, all state-of-the-art object detectors can be divided into two types: single-stage and two-stage detectors [11]. Two-stage detectors prioritize accuracy over speed, while single-stage detectors are trying to find optimal speed-accuracy trade-off [12]. Two-stage detectors are doing object detection in two sequential steps, regional proposal and classification, as shown on Figure 3. First, an algorithm, like Selective Search [13], or a convolutional neural network is used to identify regions of interest which are then passed to another convolutional neural network that performs the classification of these regions.

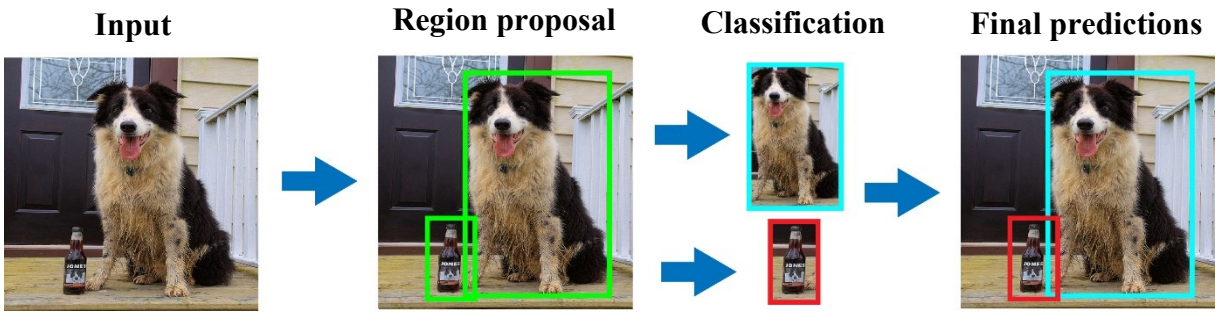


Figure 3. Two-stage detector workflow [31]. Classification is performed for every proposed region in a sequential manner.

1.2.2 Single-stage detectors

Single-stage detectors achieve their speed by performing regional proposal and classification simultaneously, as shown on Figure 4. The input image is divided into a square grid and assigns class probabilities and bounding boxes to each grid cell. Final predictions are obtained by combining results from every cell.

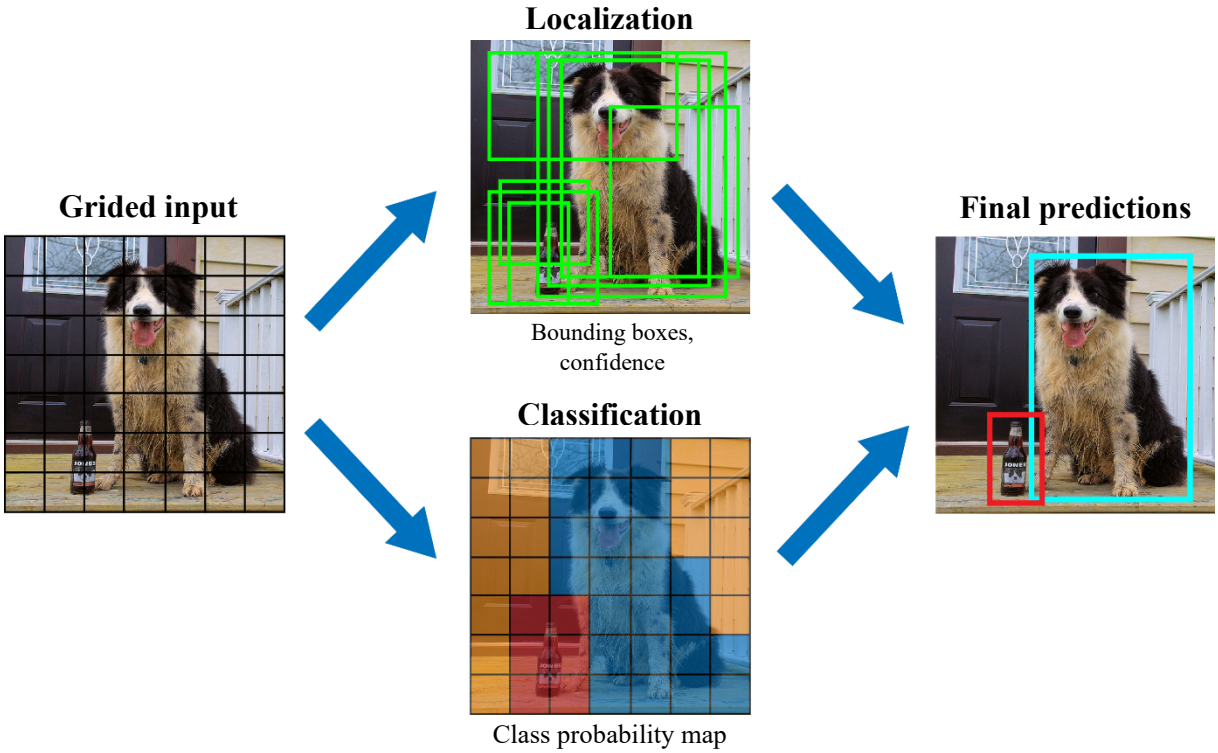


Figure 4. Single-stage detector workflow [30]. Localization and classification are performed simultaneously.

1.3 Neural network training workflow

There are three principal stages in training a neural network to perform object detection, instance segmentation and tasks that are similar to these:

1. Dataset creation
2. Training
3. Evaluation

1.3.1 Dataset creation

Creating a dataset starts with collecting images. Although the dataset should be large, this is not a defining characteristic of an ideal dataset, but rather the result of more important characteristics that greatly influence the quality of the resulting dataset. First of all, the data set should contain objects from all categories of interest in equal proportions, that is, each category should not be over- or under-represented compared to the others. Otherwise, the resulting network will be imbalanced, experiencing problems with the detection of classes of objects that were underrepresented in the training dataset as well as producing a lot of false-positive

predictions of the classes that were overrepresented in the training dataset. Secondly, objects have to be present in different environments and situations in order to represent as many real-world scenarios as possible. For instance, if car is one of the classes of interest, the dataset should contain images of cars in different weather conditions, at different times of day and night, in traffic, in parking lots, and so on. Otherwise, the resulting network will have problems detecting objects in situations other than those present in the dataset. Finally, the classes themselves should be represented by objects that differ in shape, color, and other characteristics to reflect the diversity within each class. These are not the only characteristics of the data set that affect network performance, but all others affect it to a much lesser extent.

Once all images are collected, they go through an annotation process where all objects in each image are labeled with the appropriate class and location in the image. The annotated dataset is divided into two subsets: the training set, which is used to train the network, and the validation set, which is used to evaluate the performance of the network after training.

1.3.2 Network training

Training a neural network is the process of tuning its parameters to achieve the best possible performance on a given task. Object detection neural networks can have up to hundreds of millions of parameters and it is impossible to pick the optimal ones by random guessing. To assess how well a network fits the dataset a concept called loss function is used.

Loss function maps parameter values of a neural network onto a scalar value that is called loss. Loss is the difference between annotation and network prediction. In general, the smaller it is, the better a network fits a dataset, so the goal of training can be defined as the minimization of the loss function.

It is minimized by implying an approach called gradient descent [14]. The idea of it is to randomly initialize the parameters of a neural network, produce some predictions to evaluate the loss function and then calculate a partial derivative of the loss function with respect to every parameter of a network, obtaining directions to the local minima of the loss function for every parameter. To tune the speed of the gradient descent hyperparameter called learning rate is used [15]. It scales the step, to ensure that it is not too small, so that the training will not take long, and not too big, so the local minimum is not overshot and loss will not increase.

1.3.3 Evaluation

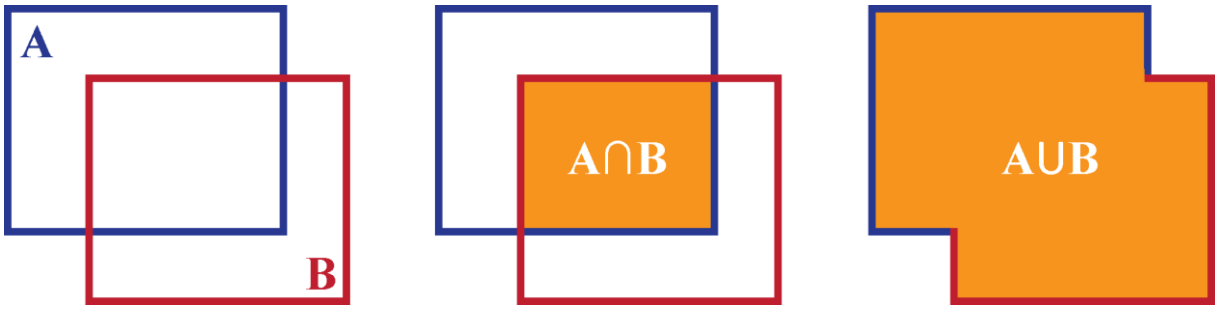
Evaluation is a process of assessment of the performance of algorithms, and in particular, neural networks. Evaluation is needed to fine-tune the parameters of the network to achieve optimal

performance. Usually, it is carried out at the end of the training, but it is also possible in the intervals between training stages, in order to obtain information about the quality and pace of the training process, to find the right moment to stop training.

There are many metrics that can be used to describe the performance of an object detection neural network. The most common of them, the ones that were used to compare the impact of different approaches designed to improve the quality of detection of small objects in this work, will be described below. But first, a few words about how the numerical data is obtained, on the basis of which these metrics are calculated. For this, a validation dataset is used. The validation set consists of annotated images, which means that all objects in these images are labeled with class and location. These images are fed into the neural network to generate predictions, which are then compared to the annotations. There are three principal outcomes when comparing network predictions with image annotations:

- *True Positive (TP)* is a situation when an object is correctly localized and classified by a network.
- *False Positive (FP)* is a situation when an object is classified or localized incorrectly.
- *False Negative (FN)* is a situation when an existing object was not detected by a network.

For correct classification, the class labels of annotation and prediction must match. The correctness of object localization is determined using a metric called intersection over union (IoU), see Figure 5. The IoU score quantifies the similarity between the predicted bounding box and the annotated bounding box.



(a) Intersecting boxes (b) Intersection (b) Union

Figure 5: Intersection and union of two boxes.

It is defined as the ratio of the area of intersection of two bounding boxes to the area of their union:

$$IoU(A,B) = \frac{|A \cap B|}{|A \cup B|} \quad (1.3.3.1)$$

The IoU score ranges from 0 to 1, where 0 is the worst case where the two bounding boxes do not overlap, and 1 is the best case where the bounding boxes match perfectly. Localization is considered successful if the IoU score is above some threshold.

The most common metrics that are calculated from the TP, FP and FN numbers are precision and recall. Precision measures the quality of predictions made by a network and is defined as the percentage of correct predictions among the overall number of predictions:

$$precision = \frac{TP}{TP + FP} \quad (1.3.3.2)$$

Recall describes the ability of a network to capture all the desired objects and is defined as the percentage of correctly detected objects from the total number of objects present in the dataset:

$$recall = \frac{TP}{TP + FN} \quad (1.3.3.3)$$

The ideal network must have high precision and recall. If the precision is high, but recall is low, a network predicts correctly, but many objects are missed. Oppositely, when the recall is high, but the precision is low, a network detects a lot of objects, but its predictions are mostly incorrect. The ratio between precision and recall can be manipulated by altering the confidence threshold. Each prediction has a corresponding confidence score which is the probability that this particular bounding box belongs to the predicted class, by introducing a confidence threshold, all the predictions with a confidence value lower than the threshold are discarded. When the confidence threshold is high, the number of predictions decreases and so does the recall, whereas the precision increases, because the number of TP decreases less than an overall number of predictions. When the confidence threshold is low situation is the opposite. Plotting recall on the x-axis and precision on the y-axis at different confidence thresholds a precision-recall curve can be built, as shown in Figure 6.

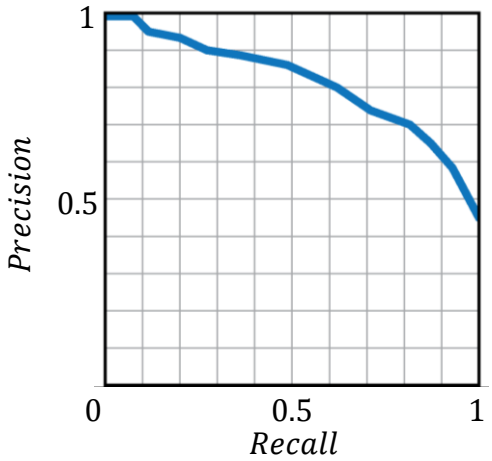


Figure 6: Precision-recall curve. Average Precision is defined as area under the curve.

On its own, this curve is not well suited for evaluating network performance, as well as for comparing different networks. That is why a metric called average precision (AP) was introduced and it is defined as the area under the precision-recall curve [16]. Another important metric is called average recall (AR) and is calculated as the doubled area under the recall-IoU curve on the interval [0.5,0.1] as shown on Figure 7. This interval is used because the correlation between detection performance and recall is achieved only for IoUs higher than 0.5. And the area is doubled for AR to take values from 0 to 1.

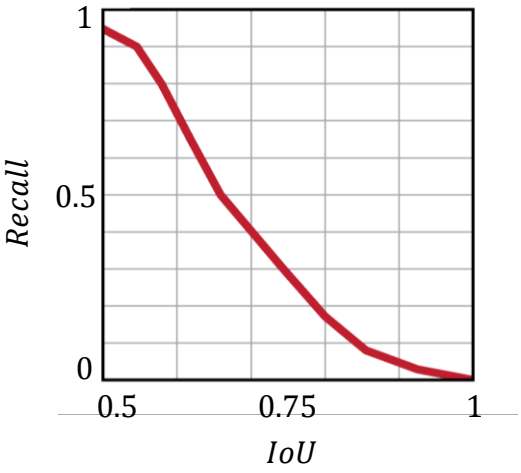


Figure 7: IoU-recall curve. Average Recall is defined as the doubled area under the curve.

For the dataset with multiple classes mean average precision (mAP) and mean average recall (mAR) are used, which are AP and AR average among all classes in a dataset, respectively.

Additionally, AP can be averaged over multiple IoU thresholds, which penalizes networks with poor localization performance.

1.4 Data augmentation

The training dataset is just as important as the architecture when it comes to model performance. However, it is not always possible to capture all of the diversity of the objects of interest found in real-world conditions. An approach called data augmentation has been developed to partially alleviate this problem [17]. This is a technique frequently used in machine learning to improve the performance of an algorithm by increasing the diversity in training data by creating copies of existing images with slight modifications done to them, see for instance Figure 8. These modifications may include flipping images horizontally or vertically, cropping, rotation, changing brightness, applying filters, making copies of objects in the image, etc. For small datasets, the previously mentioned methods will not be helpful. In such cases, new images can be synthesized from the existing ones using specific methods.

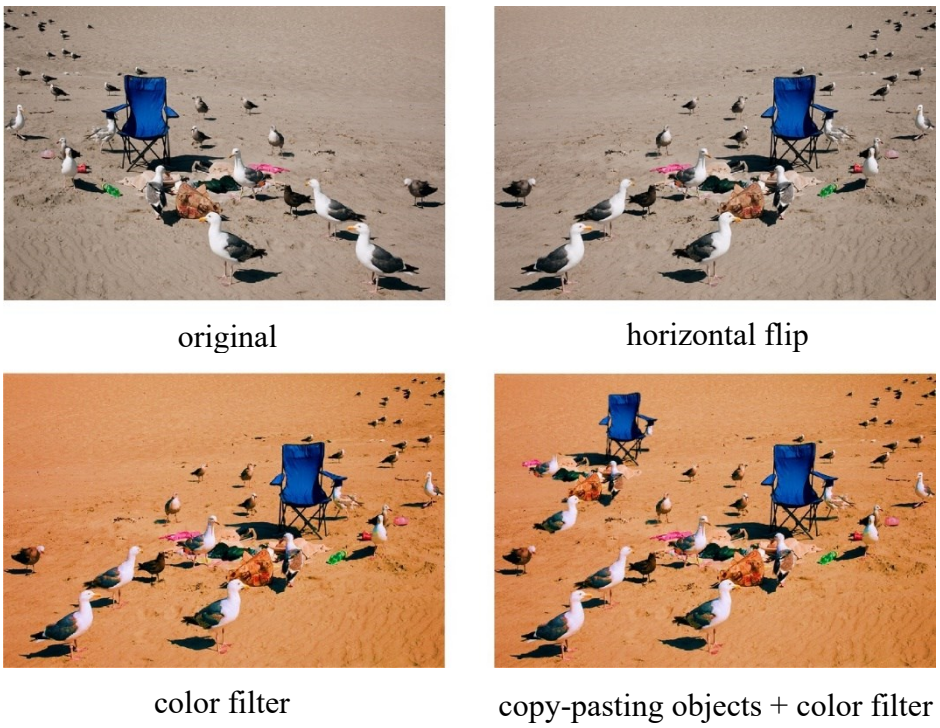


Figure 8: Examples of data augmentation. Original image is taken from MS COCO dataset.

Additionally, data augmentation can be used as an oversampling tool, to adjust the distribution of the objects in the dataset based on the class or other characteristics.

1.5 Knowledge distillation

One of the simplest ways to noticeably improve the performance of almost any machine learning algorithm is to use the same dataset to train an ensemble of different models and to use the average of their predictions as the final [18]. Although the implementation of this idea is relatively simple, it is not applicable to the problem of object detection, since convolutional neural networks are on average larger than most machine learning algorithms and require large computational resources. The alternative solution is to use a very large model that is able to achieve state-of-the-art performance, but this option is still too computationally expensive to be suitable for real-time applications such as self-driving cars. Instead, a cumbersome network or an ensemble of networks can be trained and then, using an approach called "distillation", their knowledge can be transferred to a smaller network that meets the latency and size requirements [19]. Essentially, knowledge distillation is a model compression technique in which a smaller "learner" model is trained to mimic a pre-trained larger "teacher" model [20].

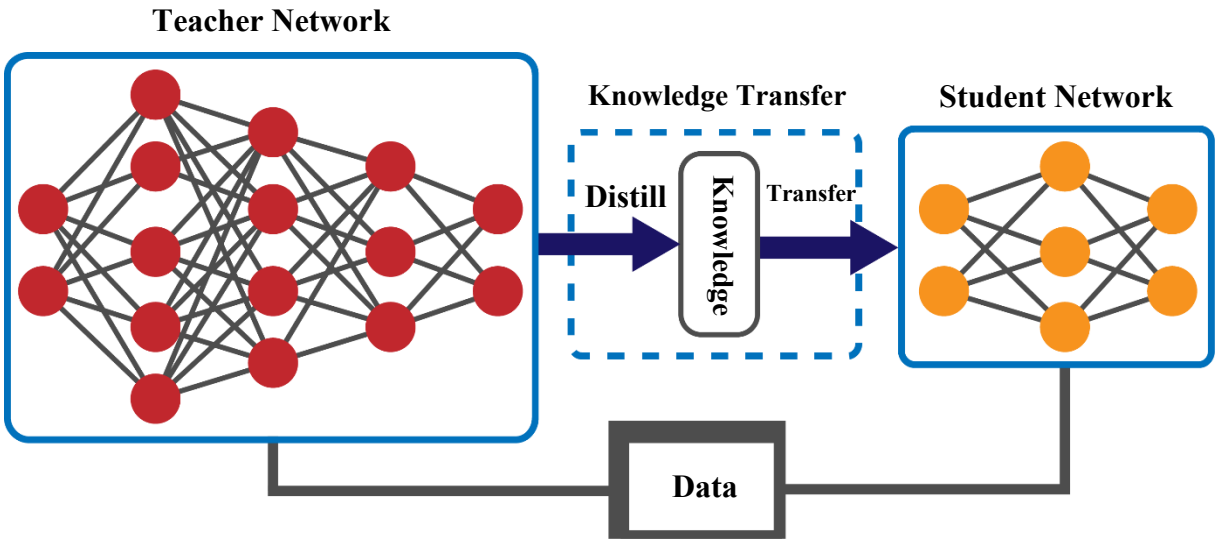


Figure 9. Knowledge distillation workflow [32].

The transfer of knowledge from the teacher network to the student is achieved by minimizing a loss function, which in this case is the difference between class probabilities distribution predicted by the teacher and student networks, which are calculated as a softmax function of logits z:

$$p_i = \frac{exp(z_i)}{\sum_j exp(z_j)}, \tag{1.5.1}$$

where p_i is the probability and z_i is the logit of class i. However, in most cases distribution of class probabilities provides information only about the most probable class, since all the other probabilities are negligibly small. This is not so much different from conventional training where the probability of an annotated class is one and all others are zero. In order to overcome this issue, Hinton et al. [19] proposed the concept of "softmax temperature". The probability p_i of each class i is calculated from the logits z as:

$$p_i = \frac{\exp(\frac{z_i}{\mathcal{T}})}{\sum_j \exp(\frac{z_j}{\mathcal{T}})}, \tag{1.5.2}$$

where $\mathcal{T}$ is the temperature.

When $\mathcal{T}$ is set to 1, the probabilities are calculated as a standard softmax function. With higher values of $\mathcal{T}$ class probabilities distribution becomes softer revealing more information about the distribution of classes other than the most probable one. In addition to the knowledge distillation, the student model is trained directly on the original dataset, minimizing the loss between its predictions and annotations. The concept introduced by Hinton et al. [19] was used as the basis for developing the channel-wise distillation method for object detection and instance segmentation by Shu et al. [21]. They proposed to softly align activation maps of corresponding channels of teacher and student networks to make better use of the knowledge in each channel. This is done by converting activation maps into a probability distribution and the channel-wise distillation loss is defined as a Kullback–Leibler (KL) divergence between the corresponding probability distributions. As demonstrated in Figure 10, the activation maps of different channels tend to encode the regions that have a high probability of being a part of an object of interest.

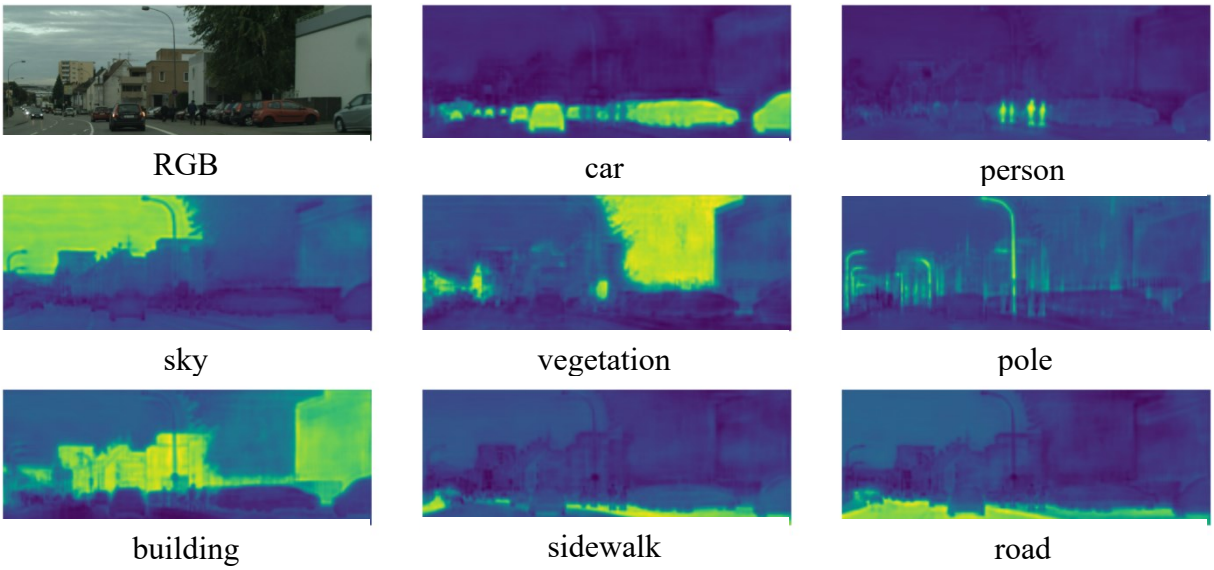


Figure 10: Activation maps of different channels. Scene saliency for each category is encoded by the activation values of corresponding channel [21].

A general formula for channel-wise distillation loss is as follows:

$$\varphi\left(\phi(y^T), \phi(y^S)\right) = \varphi\left(\phi(y_c^T), \phi(y_c^S)\right), \tag{1.5.3}$$

where y^T and y^S are the activation maps of the teacher and student networks, respectively and c is the channel index. Function $\phi(\cdot)$ converts values of activation maps into probability distribution:

$$\phi(y_c) = \frac{\exp\left(\frac{y_{c,i}}{\mathcal{T}}\right)}{\sum_{j=1}^{W \cdot H} \exp\left(\frac{y_{c,j}}{\mathcal{T}}\right)}, \tag{1.5.4}$$

where i are indexes of channels' spatial location and $\mathcal{T}$ is the softmax temperature. Function $\varphi(\cdot)$ is the KL divergence between y^T and y^S:

$$\varphi(y^T, y^S) = \frac{\mathcal{T}^2}{C} \sum_{c=1}^{C} \sum_{i=1}^{W \cdot H} \phi\left(y_{c,i}^T\right) \cdot \log\left[\frac{\phi\left(y_{c,i}^T\right)}{\phi\left(y_{c,i}^S\right)}\right]. \tag{1.5.5}$$

The KL divergence is an asymmetric metric. If $\phi\left(y_{c,i}^T\right)$ is large, to minimize the KL divergence $\phi\left(y_{c,i}^S\right)$ has to be as large too. In contrast, if $\phi(y_c^T)$ is very small, then the KL divergence pays less attention to the minimization of $\phi(y_c^T)$. Therefore, activations from the student network tend to produce a similar distribution, while activations from the teacher network's background tend to be less relevant, which is expected to have a positive effect on learning.

2 Experimental part

2.1 Dataset

For the experimental part, the Microsoft Common Objects in Context (MS COCO) dataset [2] was used. This is the state-of-the-art dataset for object detection, instance segmentation and several other image analysis applications. It has more than 200,000 labeled images, which contain 1.5 million objects of 80 different categories.

Creators of the dataset divided all images into three principal categories: iconic-object images, iconic-scene images and non-iconic images. Iconic-object images have one big object in the center of the frame that faces the camera and a relatively simple background, Figure 11 (a). Iconic-scene images are unmanned and are shot from canonical viewpoints, Figure 11(b). Non-iconic images are all images that do not fall into the first two categories, Figure 11(c). Iconic images can be easily found by image search in a browser, but they are missing non-canonical points of view and context information. The purpose of this dataset was to incorporate mostly non-iconic images, as they were proven to improve generalizing [22].

2.1.1 Training set

The original training dataset consisted of 120,000 images containing 80 object categories, but due to time and computational constraints, the dataset was reduced to 25,000 images and 2 categories: cars and humans.

2.1.2 Validation set

The original validation dataset consisted of 5,000 images containing 80 object categories. The number of images was not changed, but all object categories except cars and humans were removed to correspond to the training dataset.


(e) Iconic object images


(c) Iconic scene images


(a) Non-iconic images

Figure 11. Example of (a) iconic object images, (b) iconic scene images, and (c) non-iconic images. Non-iconic images were taken from MS COCO dataset [2].

2.2 Networks

Two object detection neural network families were used in the experiments: YOLOX [23] [24] and GFL [25] [26]. YOLOX is the family of the state-of-the-art single-stage detectors, from which the two smallest networks were used due to limited computational resources: YOLOX-tiny and YOLOX-small. GFL networks are two-stage detectors. GFL-R-50 and GFL-R-101 were chosen for the experiments.

2.3 Small objects oversampling using copy-paste data augmentation

Kisanatal et al. [27] identified two key factors that are responsible for the poor performance of small object detection: only a small portion of images in the MS COCO dataset contain small objects and small objects are not present enough even within each image containing them. The exact distributions can be seen onTable 3.

Table 3: Distribution of objects by size in MS COCO dataset [27].

	Images	Total Area
Small objects	51.82%	1.23%
Medium objects	70.07%	10.18%
Large objects	82.28%	88.59%

The proposed solution was to oversample images containing small objects and augment each of those images by copy-pasting small objects. It was proven that the best performance is achieved when only a single copy of each image with small objects is added to the original dataset, with each small object within an image copy-pasted only once, avoiding overlaps with other objects present on the image.

The approach described by Kisantal et al. [27] was implemented according to the paper and applied to the training dataset used, increasing the overall number of images from 25,000 to 30,000. As you can see from the Figure 12. objects were pasted randomly on vacant places of images, sometimes producing nonsense results.

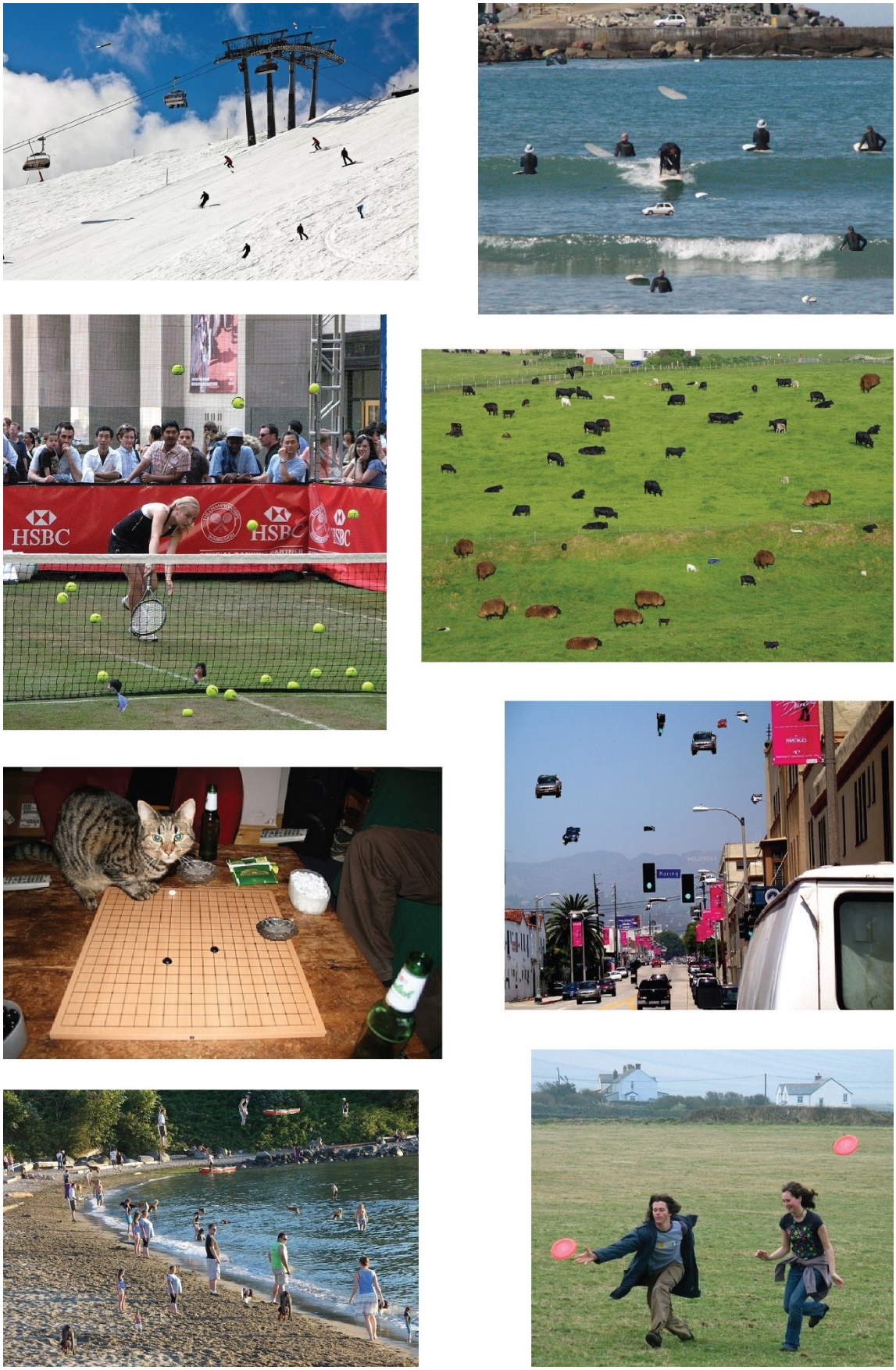


Figure 12. Examples of small objects oversampling using copy-paste augmentation.

Original images were taken from MS COCO dataset [2].

2.4 Channel-wise knowledge distillation

Training of networks using channel-wise knowledge distillation was conducted using MMRazor [28], which is an open-source model compression toolkit. It provides a framework for unified implementation and evaluation of model compression algorithms. Two datasets were used in this experiment: the original one and the dataset with copy-paste data augmentation applied to it. Each network family was trained in three different ways to explore the influence of copy-paste augmentation in more detail:

I. Teacher and knowledge distillation training were conducted on the original dataset.
II. Teacher and knowledge distillation training were conducted on the augmented dataset.
III. Teacher was trained on the original dataset and the knowledge distillation training was conducted on the augmented dataset.

In the case of YOLOX, the bigger YOLOX-small was teaching the smaller YOLOX-tiny. Similarly, GFL-R-101 was the teacher network of GFL-R-50.

2.5 Weighted loss

Another possible solution to the poor quality of small object detection is to penalize the network for the badly detected small objects during training. This can be achieved using weighted loss. The idea is to make the loss bigger, but only for small objects, so that network will pay more attention to learning to detect small objects.

YOLOX family of networks uses squared IoU loss, which is calculated by the formula:

$$IoU\_loss = 1 - IoU^2 \tag{2.5.1}$$

For medium and large objects loss formula was not modified, whereas for small objects the coefficient was added:

$$IoU\_loss_{small} = 1 - IoU^2 \cdot 0.95 \tag{2.5.2}$$

GLF network family uses GIoU loss, which is an extension of IoU loss:

$$GIoU\_loss = 1 - GIoU, \tag{2.5.3}$$

where GIoU [29] of two boxes A and B is calculated as:

$$GIoU(A,B) = \frac{|A \cap B|}{|A \cup B|} - \frac{|C \backslash (A \cup B)|}{|C|}, \tag{2.5.4}$$

where C is the smallest enclosing convex object, as shown on Figure 13.

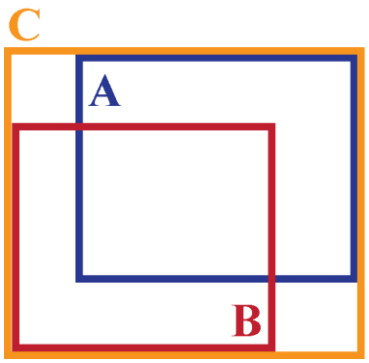


Figure 13: C is the smallest enclosing convex object for boxes A and B.

In a way similar to IoU loss in YOLOX, GIoU loss for medium and large objects remained unchanged, whereas for small objects the coefficient was added:

$$GIoU\_loss_{small} = 1 - GIoU \cdot 0.95 \quad (2.5.5)$$

3 Results and discussion

3.1 Small objects oversampling using copy-paste data augmentation

The evaluation results of the YOLOX-tiny network trained on the dataset without and with oversampling of small objects by copy-paste data augmentation (CPDA) can be seen in Table 4. The average recall for small objects has increased by almost 7%, in comparison to the unaugmented dataset. However, the change in average precision for small objects is within the margin of error, as well as average precision and recall for all other objects sizes, except for average precision for large objects which dropped by 6%.

Table 4: Comparison of YOLOX-small trained on the baseline and augmented datasets.

	AP	AP_{small}	AP_{medium}	AP_{large}	AR	AR_{small}	AR_{medium}	AR_{large}
baseline	**0.327**	0.164	**0.442**	**0.591**	0.436	0.265	**0.560**	**0.714**
CPDA	0.322	**0.168**	0.429	0.555	**0.443**	**0.283**	0.555	0.710

As can be seen in Table 5, training of GFL-R-50 on the augmented dataset has not noticeably affected the detection of small objects, but has lowered every other used performance metric, especially overall average precision, which dropped by 6%.

Table 5: Comparison of GFL-R-50 trained on the baseline and augmented datasets.

	AP	AP_{small}	AP_{medium}	AP_{large}	AR	AR_{small}	AR_{medium}	AR_{large}
baseline	**0.288**	**0.170**	**0.393**	**0.398**	**0.455**	0.322	**0.565**	**0.619**
CPDA	0.271	0.165	0.370	0.360	0.453	**0.324**	0.561	0.608

3.2 Channel-wise knowledge distillation

YOLOX training with channel-wise knowledge distillation failed, reducing each evaluation metric value many times, making the network unusable. The precise numeric results can be seen in Table 6.

Table 6: Evaluation results of the student (YOLOX-small), teacher (YOLOX-small) and the distilled student (YOLOX-small).

	AP	AP_{small}	AP_{medium}	AP_{large}	AR	AR_{small}	AR_{medium}	AR_{large}
student	**0.327**	**0.164**	**0.442**	**0.591**	**0.436**	**0.265**	**0.560**	**0.714**
teacher	0.423	0.285	0.547	0.617	0.546	0.414	0.648	0.746
distill	0.008	0.007	0.009	0.011	0.042	0.023	0.052	0.083

As can be seen in Table 7, knowledge distillation training of GFL succeeded, significantly increasing the performance of small object detection with the introduction of the augmented training set, for distillation training, and subsequently for the teacher training also. Maximal rises in small object average precision and recall of 29% and 14%, respectively, were achieved when both the teacher and distillation training were conducted on the augmented training dataset.

Table 7: Evaluation results of the student (GFL-R-50), teacher (GFL-R-101) and the distilled student (GFL-R-50) for three training datasets combinations: (I) teacher and knowledge distillation training were conducted on the original dataset; (II) teacher and knowledge distillation training were conducted on the augmented dataset; (III) teacher was trained on the original dataset and the knowledge distillation training was conducted on the augmented dataset.

		AP	AP_{small}	AP_{medium}	AP_{large}	AR	AR_{small}	AR_{medium}	AR_{large}
I	student	0.288	0.170	0.393	0.398	0.455	0.322	0.565	0.619
	teacher	0.536	0.389	0.654	0.732	0.661	0.531	0.764	0.856
	distill	0.345	0.200	**0.457**	0.524	0.487	0.344	**0.600**	**0.690**
II	student	0.288	0.170	0.393	0.398	0.455	0.322	0.565	0.619
	teacher	0.520	0.369	0.641	0.715	0.659	0.528	0.763	0.852
	distill	0.347	0.210	0.456	0.511	0.495	0.358	0.603	0.684
III	student	0.288	0.170	0.393	0.398	0.455	0.322	0.565	0.619
	teacher	0.520	0.369	0.641	0.715	0.659	0.528	0.763	0.852
	distill	**0.355**	**0.219**	0.453	**0.531**	**0.498**	**0.366**	0.597	0.688

3.3 Weighted loss

The introduction of weighted loss in YOLOX-tiny resulted in an impressive leap in detection performance of all object sizes. As can be seen from Table 8, the average precision and recall for small objects increased by 5% and 4%, respectively.

Table 8: Comparison of YOLOX-small trained with the baseline and weighted loss functions.

	AP	AP_{small}	AP_{medium}	AP_{large}	AR	AR_{small}	AR_{medium}	AR_{large}
baseline	0.327	0.164	0.442	0.591	0.436	0.265	0.560	0.714
CPDA	**0.339**	**0.173**	**0.465**	**0.599**	**0.450**	**0.276**	**0.579**	**0.729**

Despite the very good effect of the weighted loss in YOLOX-tiny, in GFL-R-50 no significant changes in the evaluation results were spotted, upon introduction of the weighted loss, as seen in Table 9.

Table 9: Comparison of GFL-R-50 trained with the baseline and weighted loss functions.

	AP	AP_{small}	AP_{medium}	AP_{large}	AR	AR_{small}	AR_{medium}	AR_{large}
baseline	**0.288**	0.170	**0.393**	**0.398**	0.455	0.322	**0.565**	0.619
CPDA	0.284	**0.172**	0.382	0.393	**0.458**	**0.329**	0.563	**0.620**

4 Conclusion and future work

All the studied approaches were able to improve the performance of small object detection, but not for all networks. Pure copy-paste data augmentation and weighted loss have not affected the quality of object detection in GFL, independent of the object size. On the other side, YOLOX appears to be not suitable for channel-wise knowledge distillation. However, the reasons for this are not clear and further investigation is needed. Weighted loss in YOLOX, instead of an expected increase in small objects metric, at a cost of the larger objects detection efficiency, unexpectedly caused noticeable performance improvement of objects of all sizes. Further experiments are required with all networks from the YOLOX family, to prove that it works not only for YOLOX-tiny. In case of success, even a 5% boost in performance, which was observed in this study, could be of great use for a large and powerful network like YOLOX-X. Also, tuning the coefficients used in weighted loss can improve results even further. Oversampling of small objects using copy-paste training data augmentation produced the weakest results among other used methods. Additionally, the obtained rise in performance of small object detection is achieved at a cost of large object detection performance, because their proportion in the dataset decreases. But still, this method proved to be effective, especially when combined with knowledge distillation, achieving an impressive 29% increase in average precision for small objects. The efficiency of channel-wise knowledge distillation was shown in its original paper, and so the results of knowledge distillation by itself were not surprising. However, the results obtained in an ensemble with copy-paste data augmentation are astonishing, not only because of the increase in small object metrics, but also because the performance of the larger objects was not significantly impaired. However, all the positive results that were shown in this study require further work of their replication on a full-scale training dataset. Nevertheless, even in their current form, the results obtained offer great promise for enhancing the detection of small objects and all the tasks that are reliant on it.

Acknowledgement

I would like to thank my supervisor, Assoc. Prof. Cagri Ozcinar, for his guidance and support throughout my thesis process. I would also like to thank Ekaterina Sedykh, who saved my thesis from failure a couple of times with her wise advice. Special thanks to Artemi Maljavin for his valuable feedback and Liina Juuse for help with translating the abstract to Estonian.

I am infinitely grateful to all the choices and coincidences in my life that saved me from doing my thesis in biotechnology.

References

[1] X. Wu, D. Sahoo and S. C. Hoi, "Recent advances in deep learning for object detection," *Neurocomputing,* vol. 396, pp. 39-64, 2020.

[2] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick and P. Dollar, "Microsoft COCO: Common Objects in Context," *arXiv:1405.0312v3,* 2015.

[3] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, O. A.-S. Ye Duan, J. Santamaría, M. A. Fadhel, M. Al-Amidie and L. Farhan, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *Journal of Big Data,* 2021.

[4] S. M. Anwar, M. Majid, A. Qayyum, M. Awais, M. Alnowami and M. K. Khan, "Medical Image Analysis using Convolutional Neural Networks: A Review," *arXiv:1709.02250v2,* 2019.

[5] A. Gupta, A. Anpalagan, L. Guan and A. S. Khwaja, "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues," *Array,* 2021.

[6] J. A. a. K. Raimond, "An Overview of Technological Revolution in Satellite Image Analysis," *Journal of Engineering Science and Technology Review,* 2016.

[7] H. Hakim and A. Fadhil, "Survey: Convolution Neural networks in Object Detection," *Journal of Physics: Conference Series,* 2021.

[8] S. Sengupta, S. Basak, P. Saikia, S. Paul, V. Tsalavoutis, F. D. Atiah, V. Ravi and R. A. P. II, "A Review of Deep Learning with Special Emphasis on Architectures, Applications and Recent Trends," *arXiv:1905.13294v3,* 2019.

[9] Z.-Q. Zhao, P. Zheng, S.-t. Xu and X. Wu, "Object Detection with Deep Learning: A Review," *arXiv:1807.05511v2,* 2019.

[10] A. M. Hafiz and G. M. Bhat, "A Survey on Instance Segmentation: State of the art," *arXiv:2007.00047,* 2020.

[11] P. Soviany and R. T. Ionescu, "Optimizing the Trade-off between Single-Stage and Two-Stage Deep Object Detectors using Image Difficulty Prediction," *arXiv:1803.08707v3,* 2018.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu and A. C. Berg, "SSD: Single Shot MultiBox Detector," *arXiv:1512.02325v5,* 2016.

[13] J. R. R. Uijlings, K. E. A. v. d. Sande, T. Gevers and A. W. M. Smeulders, "Selective Search for Object Recognition," *Springer Science,* 2013.

[14] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv:1609.04747v2,* 2017.

[15] L. N. Smith and N. Topin, "Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates," *arXiv:1708.07120v3,* 2018.

[16] M. Zhu, "Recall, precision and average precision," 2004.

[17] K. Maharana, S. Mondal and B. Nemade, "A Review: Data Pre-Processing and Data Augmentation Techniques," *Global Transitions Proceedings,* 2022.

[18] T. G. Dietterich, "Ensemble Methods in Machine Learning," Springer, 2000, pp. 1-15.

[19] G. Hinton, O. Vinyals and J. Dean, "Distilling the Knowledge in a Neural Network," *arXiv:1503.02531v1,* 2015.

[20] C. Bucilua, R. Caruana and A. Niculescu-Mizil, "Model compression," *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* pp. 535-541, 2006.

[21] C. Shu, Y. Liu, J. Gao, Z. Yan and C. Shen, "Channel-wise Knowledge Distillation for Dense Prediction," *arXiv:2011.13256,* 2021.

[22] A. Torralba and A. A. Efros, "Unbiased Look at Dataset Bias," *CVPR,* 2011.

[23] Z. Ge, S. Liu, F. Wang, Z. Li and J. Sun, "YOLOX: Exceeding YOLO Series in 2021," *arXiv:2107.08430v2,* 2021.

[24] "mmdetection implementation of YOLOX," [Online]. Available: https://github.com/open-mmlab/mmdetection/tree/master/configs/yolox.

[25] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang and J. Yang, "Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection," *arXiv:2006.04388v1,* 2020.

[26] "mmdetection implementation of GFL," [Online]. Available: https://github.com/open-mmlab/mmdetection/tree/master/configs/gfl.

[27] M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec and K. Cho, "Augmentation for small object detection," *arXiv:1902.07296v1,* 2019.

[28] MMRazor. [Online]. Available: https://github.com/open-mmlab/mmrazor.

[29] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid and S. Savarese, "Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression," *arXiv:1902.09630v2,* 2019.

[30] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *arXiv:1506.02640v5,* 2016.

[31] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *arXiv:1506.01497v3,* 2016.

[32] J. Gou, B. Yu, S. J. Maybank and D. Tao, "Knowledge Distillation: A Survey," *arXiv:2006.05525,* 2021.

NON-EXCLUSIVE LICENCE TO REPRODUCE THESIS AND MAKE THESIS PUBLIC

I, Glib Manaiev,

1. herewith grant the University of Tartu a free permit (non-exclusive license) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

Improvement of small objects detection,

supervised by Prof. Cagri Ozcinar.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons license CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in points 1 and 2.

4. I certify that granting the non-exclusive license does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Glib Manaiev

27/05/2022