The effect of grouping classes into hierarchical structures for object detection

Jordy del Castilho
Supervisors: Dr. Jan van Gemert, Dr. Osman S. Kayhan
EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Jordy del Castilho
Final project course: CSE3000 Research Project
Thesis committee: Dr. Jan van Gemert, Dr. Osman S. Kayhan, Dr. Petr Kellnhofer

Abstract

A way to reduce labelling effort and improve accuracy for object detection is class grouping. In this research, we experiment with creating hierarchical tree structures of grouped classes (super-classes). Our objective is to find out what the effects are of grouping classes in terms of accuracy and labelling effort. First we show what accuracy improvement can be gained from different grouping strategies. Then we show what the difference is in accuracy gain for a hierarchical tree structure on FasterRCNN, RetinaNet and YOLOv8. Next up we introduce a new layer in the tree structure with new super-classes and we show the difference in the tree with two and with three layers. After that we compare the accuracy for predicting only super-classes. Lastly, we want to take predicting super-classes one step further by showing that class grouping can reduce labelling effort for real life applications like autonomous cars.

1. Introduction

Object detectors are used for more and more applications every year and these detectors need training data [8]. Training data is sometimes scarce because labeling images takes lots of effort. Labelling requires the annotator to accurately put bounding boxes on all objects from the wanted classes and annotate them with the correct class. To get the most out of the effort spent, labelling must become easier and object detection needs to become more accurate without requiring more labeled data.

To tackle these problems, we are using a recently researched tactic [5] of creating super-classes by grouping visually/semantically similar classes and forming a hierarchical structure. For example the classes cat, dog, sheep and cow can be grouped to two super-classes "cat or dog" and "sheep or cow". This grouping is done to help the object detector focus on learning differences between the grouped classes.

This paper addresses the following two research questions: 1. Are we able to increase model classification accuracy without extra labeling effort? 2. Are we able to reduce labeling effort while keeping at least the same accuracy? These questions are answered through multiple experiments.

2. Related research

2.1. Object detectors

In 2014 an object detector called RCNN (Region based Convolutional Neural Network) was created by Ross Girshick [11]. Despite being revolutionary, this model was slow and frail and was therefore immediately updated by the creator and named Fast-RCNN. In 2015 FasterRCNN [10] was created and released. This model was faster than the previous 2 versions. In our research we mainly use Faster-RCNN for the experiments. This model is a popular base model for research because it is complex and customisable and in terms of robustness it has not been outperformed by many newer models. In 2015 YOLOv1 was introduced [6]. unlike many other object detectors, YOLOv1 is fast and has a simple framework but is not easily customisable. In one of our experiments we used YOLOv8, which is the 8th version of the YOLO object detectors. In the experiment we use YOLOv8, we also use RetinaNet [12]. We have chosen these three models because they are all popular models used for research.

2.2. Class grouping

In this research we have conducted experiments that use class groupings and hierarchical structures. Earlier research related to this can be found in the paper by Tian et al. [8]. In their experiments, hierarchical structures are used with modified detection heads, which are the parts of the object detector that do the predictions of bounding boxes and classes. These detection heads were specifically trained for coarse-grained and fine-grained differences. In our research, however, we focussed solely on the effect of grouping classes in terms of accuracy and labeling effort. A more similar paper to ours, written by Fard et al. [5] tried to tackle this problem in a similar approach to ours. They built a new object detector from scratch in such a way that it hierarchically trains the model. The author explains why the classes that should be grouped together should be the most similar classes for the most increase in accuracy. The author proposes to do this is by using the confusion matrix from a normal model. However, we have decided to take an alternative approach because in practice there is never access to a confusion matrix before the model has been trained. We have chosen to group classes based on semantic and visual similarities instead.

3. Methodology, Experiments and results

The experiments were conducted using Detectron2 [15], which is a framework developed by Meta for object detection research, and YOLO [2]. The dataset we used for most experiments is the Pascal VOC 2012 dataset [1]. This dataset is public and has 20 classes. For our experiments, we have chosen 10 of these classes in such a way that there will be 5 groups of visually and semantically similar classes. These groups being: "Cat or Dog", "Cow or Sheep", "Motorbike or Bicycle", "Bus or Car", "Sofa or Chair". With these 10 classes the first 4 experiments were conducted. A last experiment is conducted with a dataset containing car dash-cam images. For the experiments using Pascal VOC 2012, the included test set was used for evaluation.

For accuracy evaluation we chose to measure only the classification accuracy. We chose this because it isolates the classification, which is the part that is affected by these experiments. We first make sure our test set contains only images with 1 annotation by splitting images with multiple annotations. Then we run inference over all those images and calculate the classification accuracy of correct class predictions.

$$\text{Classification accuracy} = \frac{\text{Amount of correct predictions}}{\text{Amount of predictions}}$$

For labeling reduction we evaluate the difference in the amount of classes. When two classes can be grouped the labeling effort is reduced by 1 class since the difference between the initial state and the states where the classes are grouped is 1.

In the hierarchical tree structures each node is a model. We chose this because it did not require us to internally modify the architecture and therefore was the easiest for our limited time frame.

All nodes of the tree and the baseline model are trained from the pre-trained weights on the Common Objects in Context (COCO) dataset [13]. The nodes were trained for 5000 iterations and with a learning rate of 0.0005 that gets divided by 10 at 60% and 80% of the iterations. Furthermore, a linear warm up method was used with a factor of 1/1000 for 1000 iterations. Lastly a batch size of 2 was used.

To predict within the tree, we run inference on the root node and based on the prediction we walk down the tree and run inference on the corresponding node.

In all experiments FasterRCNN was used as model. This choice was made because FasterRCNN is a popular model choice for research and could also be implemented and configured easily using Detectron2. In the second experiment, YOLOv8 and RetinaNet were also used.

In the first 3 experiments, we have focused on answering our first research question by showing accuracy improvements for hierarchical structures. In the last 2 experiments we have focused on the second research question by showing how labelling effort can be reduced by class grouping.

3.1. baseline versus semantically grouped versus randomly grouped.

3.1.1 Setup

For this experiment, we have trained a baseline model for comparison. Then we have trained the tree structure with the semantically grouped classes. For this we iterate over every node in the tree and train the corresponding model. The root node will be trained on all the images from the 10 classes. It will train using the 5 groups as classes. This is illustrated in figure 1. Next up all the children nodes of the root are trained. Each child node will be a model which is trained to only make a distinction between 2 classes. This is also illustrated in figure 1. Now we have trained the semantically grouped tree. We proceed to train the randomly grouped tree in the same way. To randomly group classes we can use a random number generator from 1 to 10 and make pairs each 2 numbers that have not been used already. By doing that, we got the randomly selected groups: sofa and sheep, cat and cow, bicycle and car, dog and motorbike, bus and chair. A visual representation of this tree can be found in figure 2.

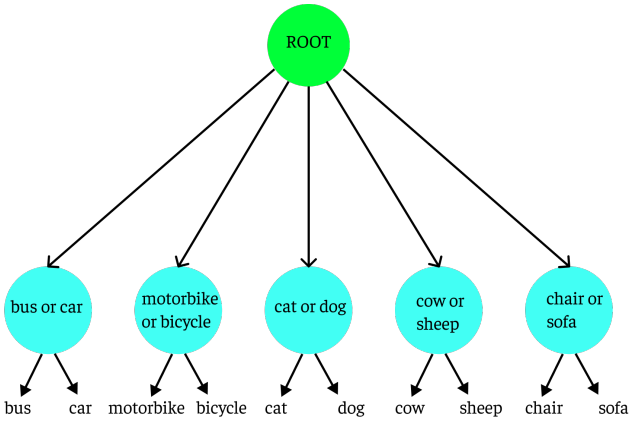

Figure 1. Shows the tree structure used for experiments 1,2 and 3. All groups are chosen based on semantic/visual similarities for the best results. Each node in the tree represents a model. The root node makes a prediction between the 5 super-classes it was trained on. All children nodes do binary classification and are only trained on the images of the 2 classes they classify

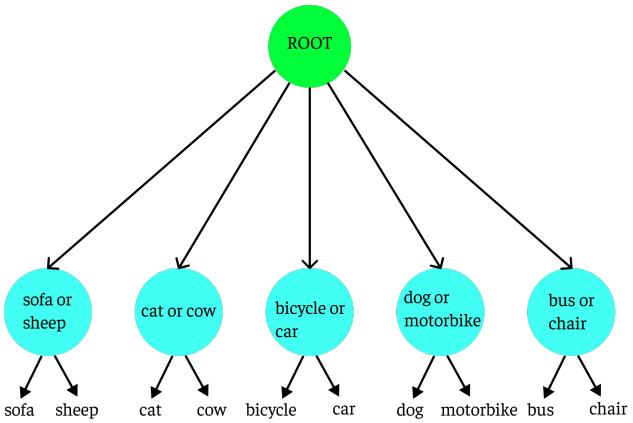

Figure 2. Shows the tree structure used for experiment 1 when the grouping of classes is done randomly instead of semantically.

We have conducted this experiment to investigate the effect of the grouping strategy in a hierarchical structure.

3.1.2 Results

From this experiment we have got some interesting results (Figure 3). We can see that we can increase the accuracy of the model by 1.6% by using semantically grouped classes within the tree structure. We can also see that the grouping choices do matter as randomly grouped has a lower accuracy than the baseline model.

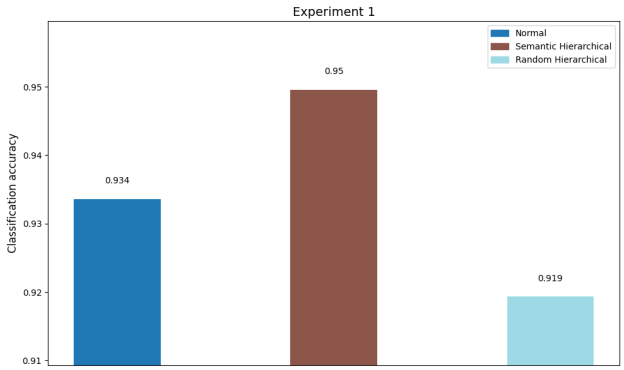

Figure 3. Comparison of classification accuracy on the test set containing only 1 label per image of the Baseline model (Normal), the tree structure using semantically grouped classes (see figure 1) and the tree structure using randomly grouped classes (see figure 2). This shows that the grouping strategy matters and when visually similar classes are grouped together the tree has a higher classification accuracy because it specifies to learn the details.

3.2. Hierarchical Comparison between FasterRCNN, YOLO and RetinaNet

3.2.1 Setup

For this experiment, we use the same semantically grouped structure (figure 1) for all models and compare if the effect of grouping classes is different for these models. RetinaNet and FasterRCNN could easily be implemented within Detectron2, we have only had to change the model and weights parameter of the configuration. RetinaNet and FasterRCNN were both trained for 5000 iterations in each node. For YOLOv8 we had to use the YOLO framework from Ultralytics. The same learning rate was used for the YOLO experiment. However, the amount of iterations was set to 75000 since the initial results with 5000 iterations were not that accurate. YOLO expects a number of epochs but Detectron2 expects a number of iterations so to solve this problem we used the following formula:

$$\text{amount of epochs} = \left\lfloor \frac{\text{amount of iterations}}{\text{amount of images}} \right\rfloor + 1$$

This experiment was conducted to investigate if not only fasterRCNN but also other models with different architectures benefit from class groupings and hierarchical structures. We do not care about the accuracy of the different models. Instead we care about the accuracy gain from converting to the hierarchical counterpart of the different models.

3.2.2 Results

The results are shown in figure 4. The FasterRCNN result was copied from the last experiment but we introduce the accuracy gain for the object detectors RetinaNet and YOLOv8 as well in here. RetinaNet has an accuracy increase of about 2.9% and therefore has almost double the increase of fasterRCNN. YOLOv8 has an increase of about 2.2%.

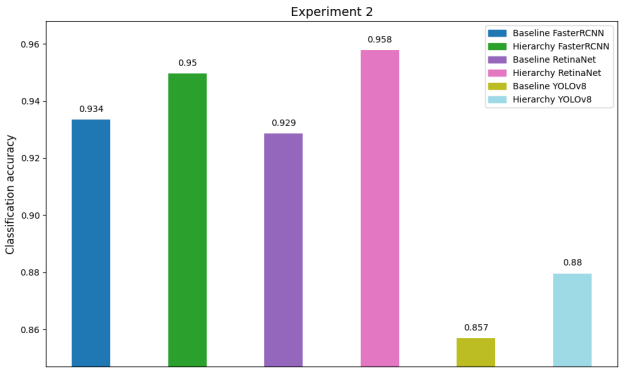

Figure 4. Shows the accuracy gain for different detectors when comparing baseline models to their hierarchical counterpart using semantically grouped super-classes (figure 1). The fasterRCNN case of this figure is the same result used in the previous subsection but was shown here as comparison. This shows that not only fasterRCNN but also YOLOv8 and RetinaNet can benefit from these class groupings and hierarchical tree structures.

3.3. 2-layered hierarchy versus 3-layered hierarchy

3.3.1 Setup

For this experiment, we took the semantically grouped tree and modified it to have 2 extra nodes such that the root node only predicts between 3 different nodes. The first new node will contain all animals. The second new node will contain all vehicles. This is illustrated in figure 5.

The reason for this experiment is to find out if we can reach a higher accuracy with an extra layer within the hierarchical tree structure.

3.3.2 Results

In the results shown in figure 6 we see that the extra internal nodes do help the accuracy of the structure because there is almost a 1% increase in classification accuracy.

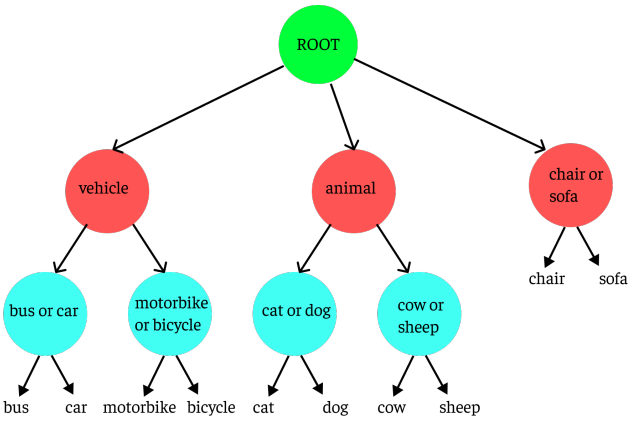

Figure 5. Shows the tree structure used for experiment 3. This depicts the tree structure with extra super-class nodes added such that the root only predicts between 3 classes rather than 5.

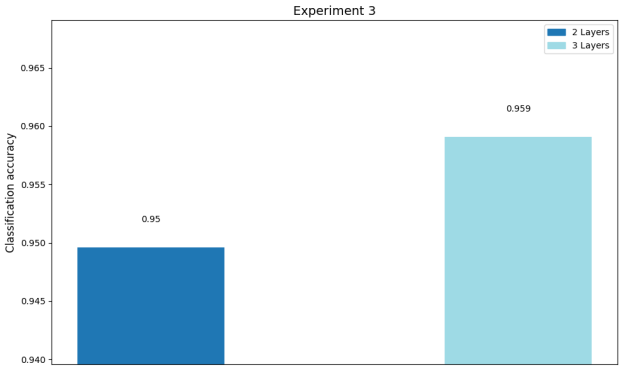

Figure 6. Shows the difference in classification accuracy on the test set between a tree structure with 2 layers (the root node in the first layer and 5 children in the second layer as shown in figure 1) and a tree structure with 3 layers (the root node in the first layer. 3 nodes in the second layer and 4 more nodes in the third layer as shown in figure 5). From this we can conclude that adding more layers in the tree with extra super-classes can also improve performance.

3.4. Predicting super-classes versus predicting base classes

3.4.1 Setup

For this experiment, we compare the accuracy of predicting super-classes. For a hierarchical tree structure predicting super-classes is what a node without leaf nodes as children does. For this experiment we look at the root nodes of figure 1 and figure 5. For a normal model, predicting super-classes means that the model predicts a class which is part of the super class. This means that in a scenario where cat and dog are grouped, for the normal model to predict cat means it predicts the super-class ”cat or dog”. So even if the ground truth label was dog it still predicted the

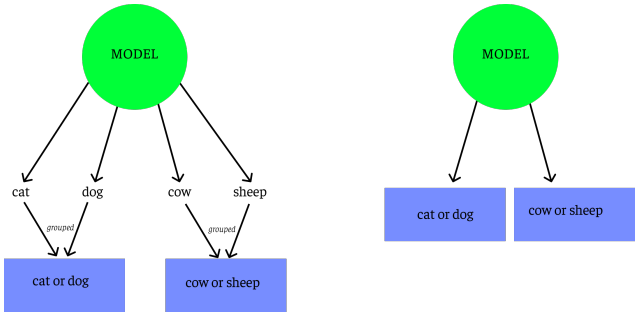

Figure 7. Visual explanation for structures used in experiments 4 and 5. This example is only for 4 classes: cat, dog, cow and sheep. On the left you have the base model, which predicts out of all classes and the classes are combined into a group afterwards. When the ground truth is cat, anything that is predicted as cat or dog is counted as correct group prediction. On the right we have the model which is trained on the super-classes instead. These two models have the same training time, same configuration and same training dataset. The only difference is that the annotations of the left model distinct between 4 classes while the annotations for the right model are grouped such that there are only 2 classes.

super-class correctly. A visual example for this is shown in figure 7. This can be useful in many cases where object detection is used to perform an action based on the detected class. When the detector is trained on more classes than actions according to the pigeon hole principle [9], at least 2 classes detected will lead to the same action. If a model trained on grouped classes achieves higher accuracy when predicting super-classes than the baseline model trained on base classes then it is beneficial to retrain the model on grouped classes. Furthermore, the results can introduce new best practices for object detection standards which might feel unnatural at first. To give an example for this specific problem we can take an autonomous car. for simplicity we can assume 2 actions, drive or stop. traditionally object detectors used in autonomous cars may predict a set of many classes like stop sign, red-light, green-light, pedestrians, shark teeth and crosswalks. Instead of training our object detector on all these classes we can try to train it on 2 groups, the objects that would lead to stopping and the objects that would lead to driving. Because of this new best practice for training data, we essentially reduce the amount of classes for the annotator and thus reduce the labelling effort by getting the same desired results but less classes. For this experiment we want to test this on the Pascal VOC 2012 dataset with the same groups as the root nodes from the trees from figure 1 and figure 5. In the next experiment we have tried this on a dataset specifically for autonomous cars.

3.4.2 Results

The results in figure 8 show that predicting super-classes has a higher classification accuracy for models that were trained on these super-classes than models that were trained on the base classes. This means that predicting classes like ”animal” or ”vehicle” for the respective super-classes is more accurate on a model where all animal base classes are grouped and all vehicle base classes are grouped. The figure shows that this is the case for both the 2-layered tree and the 3-layered tree.

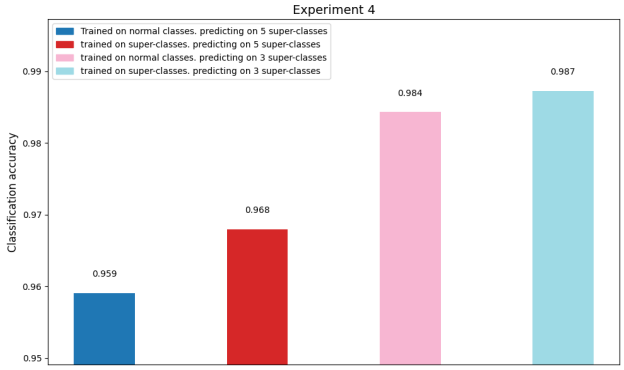

Figure 8. Shows the difference between predicting super-classes by models trained on super-classes and models trained on base classes. This was analysed with the root node predictions of the structure from (figure 1) with 5 super-classes and the root node predictions of the structure from (figure 5) with 3 super-classes. This shows that the models which were specifically trained on super-classes perform better on those super-classes than the models which were trained on all base classes.

3.5. Reducing labeling effort for autonomous cars

Figure 9. Shows how different class labels are grouped together such that each action of the application is done based on only 1 prediction of a super-class. Picture is taken from dataset used in the experiment but manually labelled to include extra classes to show as example. Left is what the labelling would look like if base classes were used. Right is what the labelling would look like if the classes were grouped.

3.5.1 Setup

For this experiment we repeat the same experiment as 4 but on a different dataset. We found a dataset on Roboflow, a platform for object detection training data creation and modification. This dataset has 11 classes. For simplicity we took a subset of this data. We have taken 5 classes and grouped them into driving and stopping super-classes (Figure 9). in the driving group we have the classes traffic-light-green and traffic-light-yellow. For the stopping group we have traffic-light-red, pedestrian and biker. Furthermore, We only take the first 10000 images of this set and split into 70% training and 30% test. We train two models, one of which predicts between the groups stopping and driving. The other will predict between all 5 classes and will return the action based on the class predicted. For this experiment we have one case where the 2 models were trained for 5000 iterations and another case where they were trained for 25000 iterations.

3.5.2 Results

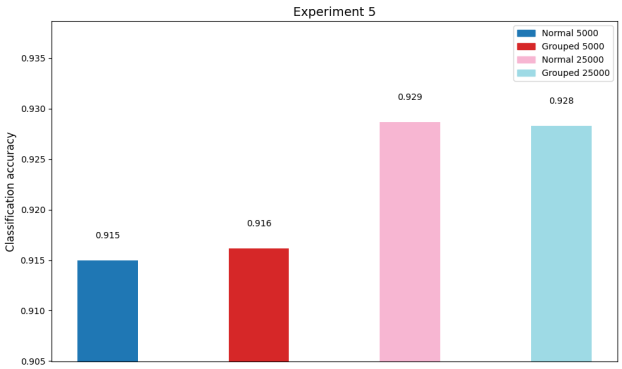

Figure 10. Comparison between a post-classification-grouped model that predicts all 5 classes and a model that predicts only the 2 super-classes both for 5000 and 25000 iterations on a dataset used for autonomous car object detection. These results show that there is no difference in predicting the super-classes driving and stopping. These super-classes were formed using the base classes green-traffic-light, yellow-traffic-light for the driving super-class and biker, pedestrian, red-traffic-light for the stopping super-class. Even though red traffic lights, green traffic lights and yellow traffic lights are very similar, this still shows there is no loss in classification accuracy. Furthermore, it shows that labeling effort can be reduced. This dataset can be modified to only have 2 super-classes and new data can be labeled with those two super-classes to reduce the effort for the annotator.

The results of this experiment can be found in figure 10. Looking at the results we see a very slight accuracy increase in the 5000 case but also a very slight accuracy decrease in the 25000 case. Since these increases and decreases are

only very little, we think it is safe to say that the accuracies are about the same. Despite not reaching higher accuracy in this experiment we did get a desired result because we reduced the labelling effort. In detail: we decreased the amount of classes from 5 (red traffic light, green traffic light, yellow traffic light, biker, pedestrian) to only 2 (driving, stopping). Since super-class prediction is only training 1 model we also did not increase the training time. The accuracies probably did not improve because the classes that were grouped were not that visually similar. This is shown in experiment 1 as well.

4. Responsible Research

4.1. Reproducibility

To ensure reproducibility, the FAIR (Findable, Accessible, Interoperable, Reusable) principle was used [14].

4.1.1 Findable

The main code as well as the supplementary code is available on GitHub. Together with the code and the training specifications mentioned in the methodology section another researcher can reproduce the results.

4.1.2 Accessible

The GitHub repository is made publicly accessible for everyone. The datasets used are also publicly accessible for everyone. The Detectron2 and YOLO frameworks are also publicly available for personal and research purposes. All these things require no authentication.

4.1.3 Interoperable

The code has been written in Python, which is the standard for machine learning research. The libraries used are commonly used in computer vision tasks. Integration with this code is easy because the Detectron2 framework, which is specifically made for research, is used. Furthermore, the code includes comments which show the steps taken to get to the result.

4.1.4 Reusable

The code is commented and can be reused by other researchers. When used on the same datasets as described in this paper, the experiment can be replicated. The metadata is also described within the text and the figures of this paper.

4.2. Integrity

This research adhered to five principles of research integrity defined in the Netherlands Code of Conduct for Research Integrity [3]. These principles being: Honesty, scrupulousness, transparency, independence and responsibility. All research is presented honestly and with careful attention to detail. Full transparency is ensured by providing the experiment code and parameter configurations.

5. Limitations

For this research, there are several limitations that should be addressed. First of all, all models are not trained from scratch but from pre-trained weights. This choice was made to speed up the research process. Second of all, within this research we have not modified the architecture of the models. This means that for creating the hierarchical structure, some model parameters have been retrained unnecessarily. Lastly, the results shown in this paper have not been aggregated from multiple instances of the same experiment. Despite this being the case, the experiments have been checked with running the experiment a second time. The second time the experiments were run was only to verify that the results from the first experiment were accurate and reproducible and therefor were not used in the results of this paper.

6. Conclusion and future work

6.1. Conclusion

The main research questions of this paper were: 1. Are we able to increase model classification accuracy without extra labelling effort? 2. Are we able to reduce labelling effort while keeping at least the same accuracy? This paper has shown how classification accuracies of existing object detectors can be improved without extra labelling in the results of experiments 1,2 and 3. In these experiments, it has been shown that object detectors can benefit from hierarchical structures when using a good grouping strategy. This can be used for use cases where resources are irrelevant and the model cannot reach higher accuracy anymore from training on the same data. Next up this research answers the second research question with experiment 4 and 5, explaining that models trained on super-classes have a better accuracy on those super-classes than a model trained on all classes. This can be very valuable in use cases where certain actions are taken based on classes predicted but the amount of actions is less than the amount of classes. By showing this we conclude that in many cases the labelling effort can be reduced by only having the needed super-classes instead of the initial base classes. That way the labeller only has to choose from a smaller set of classes when annotating the data. In experiment 5 we have taken this idea to another level by making a simplified version that would improve the labelling effort while not increasing resources and not losing accuracy for automatic driving cars. This could introduce a new best-practice for object detection which en-

courages people to group classes together when their application does not discriminate further between a set of predicted classes.

6.1.1 Extra note

In the results of experiment 2 our results show that YOLOv8 can benefit from hierarchical structures. However according to Tian et al., "most existing object detection algorithms such as SSD, YOLO and its variants, Mask RCNN are designed to exploit labels in a flat or uniform manner. Consequently, these models cannot benefit from hierarchical nature of the data. [8]" Our experiment disproves this statement. Another interesting result we see in experiment 5. In experiment 1 we have shown that grouping visually similar classes together increases the accuracy while grouping randomly decreases the accuracy. This is what also happens in experiment 5. Since the red traffic light class which is more visually similar to the yellow and green traffic light was not put in the same group as them the accuracy likely dropped. Despite this we still managed to get the same accuracy as when the model was trained on all base classes so the experiment still worked out.

6.2. Future work

There still is a lot of research to be done about this topic. The hierarchical structure used in this paper is not compact or fast to train so future work could include optimising this structure. Furthermore, more research can be done with different datasets like COCO MS and also autonomous car datasets like Kitti [4], different configurations and different architectures. We further propose that more instances should be found where predicting super-classes is just as accurate or more accurate when the model has been trained on these super-classes. Like the autonomous car instances this can be applied in many other cases like surveillance cameras, where everything that should raise an alarm can be grouped, or farming scenarios, where all types of weeds can form a super-class and all types of crops another super-class. Next up there can be experimented with modified models where the classification is not a chosen base class but a location in hyperbolic space [7]. based on this location the class can be predicted. By using hyperbolic spaces with 2 dimensions we can more easily map out hierarchies with equal distances between the nodes such that the model can learn more accurately to distinguish between classes.

References

[1] Pascal voc 2012. 1

[2] Ultralytics yolo. 1

[3] *Netherlands Code of Conduct for Research Integrity*. Association of Universities in the Netherlands (VSNU), 2018. Translated version of the "Nederlandse Gedragscode Wetenschappelijke Integriteit (2018)". Licensed under CC-BY 4.0. 6

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 7

[5] Quoc-cuong Pham Antoine Vacavant Hamidreza Odabai Fard, Mohamed Chaouch and Thierry Chateau. Joint learning for multi-class object detection. *CEA, LIST, Vision and Content Engineering Laboratory, Point Courrier 94, F-91191, Gif-sur-Yvette, France Institut Pascal, UMR6602, CNRS, Blaise Pascal University, Clermont-Ferrand, France*, 2014. 1

[6] Ross Girshick-Ali Farhadi Joseph Redmon, Santosh Divvala†. You only look once: Unified, real-time object detection. *Facebook AI Research (FAIR)*, 2015. 1

[7] Tamara Munzner. Hyperbolic space, 1997. [Online; accessed july 20, 2024]. 7

[8] Amit Kumar K C Qing Tian, Sampath Chanda and Douglas Gray. Improving apparel detection with category grouping and multi-grained branches. *ECE Department, McGill University, Montreal, QC, Canada Amazon Visual Search AR, Palo Alto, CA, USA*, 2021. 1, 7

[9] Alexander A. Razborov. Proof complexity of pigeonhole principles. *Steklov Mathematical Institute, Moscow, Russia Institute for Advanced Study, Princeton, USA*, 2002. 4

[10] Ross Girshick Jian Sun Shaoqing Ren, Kaiming He. Faster r-cnn: Towards real-time object detection with region proposal networks. 2015. 1

[11] Shiksha. Object detection using rcnn, 2024. Accessed: 2024-06-11. 1

[12] Ross Girshick Kaiming He Tsung-Yi Lin, Priya Goyal and Piotr Dollar. Focal loss for dense object detection. *Facebook AI Research (FAIR)*, 2018. 1

[13] Serge Belongie Lubomir Bourdev Ross Girshick James Hays Pietro Perona Deva Ramanan C. Lawrence Zitnick Piotr Dollar Tsung-Yi Lin, Michael Maire. Microsoft coco: Common objects in context. *arXiv:1405.0312v3*, 2015. 2

[14] Mark D. Wilkinson et al. The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3: 160018–, 2016. 6

[15] Francisco Massa Wan-Yen Lo Ross Girshick et all Yuxin Wu, Alexander Kirillov. Detectron2. *Facebook AI Research (FAIR)*, 2019. 1