Contents

[1.0 Introduction 2](#_Toc137139815)

[1.1 Background 2](#_Toc137139816)

[1.2 Dangerous Goods Labels 2](#_Toc137139817)

[1.1 Problem Statement 3](#_Toc137139818)

[1.3 Aim 3](#_Toc137139819)

[1.4 Objectives 4](#_Toc137139820)

[2.0 Methodology 5](#_Toc137139821)

[2.1 Data 5](#_Toc137139822)

[2.2 Data Extraction 5](#_Toc137139823)

[2.3 Data Pre-processing 6](#_Toc137139824)

[2.4 The Deep Learning Model – YOLO 8](#_Toc137139825)

[2.4.1 Training the Deep Learning Model 9](#_Toc137139826)

[3.0 Results and Discussion 11](#_Toc137139827)

[3.1 Training Results 11](#_Toc137139828)

[3.2 Test Results 12](#_Toc137139829)

[3.3 Deployment 15](#_Toc137139830)

[3.4 Recommendations and next steps: 15](#_Toc137139831)

[4.0 Conclusion 16](#_Toc137139832)

[5.0 References 17](#_Toc137139833)

# 1.0 Introduction

# 1.1 Background

Lübecker Hafen-Gesellschaft mbH (LHG) is the company responsible for managing the publicly operated ports in the Hanseatic city of Lübeck, specifically the port of Lübeck. LHG is the largest seaport operator in Germany and operates the country's largest container port situated on the Baltic Sea.

LHG operates four port terminals along the Trave River Lübeck with modern handling facilities and highly qualified personnel.In addition to handling Roll-on/Roll-of (RoRo) traffic, the terminal also offers weather-protected storage, loading of forestry products.

At the entrances and exits of the Skandinavienkai RoRo terminal, all units i.e. trucks moving cargo into or from the terminal are guided through a scanning system. During the scanning process, several pieces of information are captured. They include licence plates and length measurements. For this purpose, high-resolution images of all sides are taken of the units (trucks).

In addition to the above data, dangerous goods labels attached to different locations on the trailer are automatically detected. However, no classification is made based on the content of the dangerous goods label that tells anything about what type of dangerous goods are in the trailer.

# 1.2 Dangerous Goods Labels

Dangerous goods labels are labels used to identify the hazard class or division of a dangerous good when transported by air, land or water. They are the primary means of identifying dangerous goods by port terminal staff. In the EU there are strict rules regarding the transportation of dangerous goods. These rules are based on the UN Recommendations on the Transport of Dangerous Goods [[1]](https://unece.org/fileadmin/DAM/trans/danger/publi/unrec/rev21/ST-SG-AC10-1r21e_Vol1_WEB.pdf). One important aspect of the rules is the correct labelling of units conveying dangerous goods. The figure below shows the different categories of dangerous goods labels.



*Fig 1.1 Dangerous goods labels*

## Problem Statement

Although LHG captures images of all sides of trucks moving into and out of the terminal, the task of identifying the actual label class and confirming whether this matches the information in the electronic booking data sent in advance by the shipping companies is still manual i.e., staff need to look at the images to verify this information.

## 1.3 Aim

To create an automated system for identifying and verifying the correctness of dangerous goods labels on trucks moving into and out of the Skandinavienkai terminal.

## 1.4 Objectives

The major objective of this project is:

1. The implementation of a Machine Learning solution for the identification and classification of dangerous goods labels.
2. Depending on the availability of time, the following additional objectives are to be addressed:

* Integration and comparison with existing booking data from the LHG logistics system.
* Display of deviations in label and booking data in a web frontend
* Implementation of a push mechanism (e-mail) for notification of deviations

# 2.0 Methodology

## 2.1 Data

The LHG provided the data used for this project: more than 6 terabytes of data representing images of trucks, taken by the scanning system. The data is organised into four folders, one for each of the four gates (two at the entrance and two at the exit). Each folder contains 236 subfolders, each corresponding to a day of the year. In each of these daily folders, there are several zip folders, one for each truck that passes through the scanning system. Each zip folder contains an average of 40 images taken from different angles and of different sides of the trucks. The structure of the data folder can be summarised as follows:

* Folder with the name of the gate
  + Folders for each day (365 folders)
    - Zip folders for each vehicle scanned
      * Images of different sides of the vehicle taken from different an gles (about 40 photos on average)

## 2.2 Data Extraction

After careful sample checks and analysis, it was concluded that images of the rear of the trucks would be most suitable. This is because, as per regulation, a truck bearing dangerous goods must always have a sign at the rear of the truck. Images of the rear follow a naming convention which makes their identification relatively easy. For example: “snapshot-chassis0-Isback-full.jpg”

These images were extracted using a Python scripten. A total of 547,000 images were extracted. These were further reduced to about 338,000 after excluding non-relevant images such as snapshots of licence plates.

## 2.3 Data Pre-processing

In order to run the machine learning algorithm used for this project on the images, images had to be first annotated. Annotation of images in the context of this project is the drawing of a bounding box around the dangerous goods labels in an image and labelling the box with the correct label class.

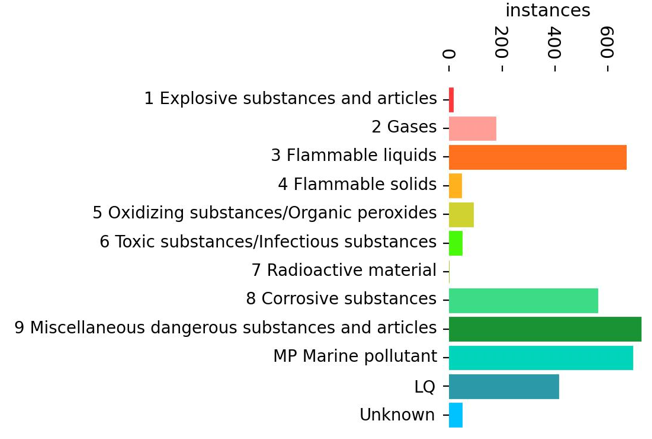


*Fig 2.1 Annotated image of a truck with a ‘corrosive’ label.*

Labelling of the images was done using [MakeSense.ai](https://www.makesense.ai/). It is a free online labelling tool that is:

* Open source and free to use under GPLv3 license
* Requires no advanced installation. Accessible via a web browser.
* Does not store the images nor transmit images.
* Supports multiple label types - rectangles, lines, points and polygons (in this project we use rectangles)
* Supports various output file formats like YOLO, VOC XML, VGG JSON, and CSV (in this project we use rectangles YOLO)

A total of about 70,000 images were processed via the web labelling tool and visually inspected for the presence of dangerous goods labels. This resulted in about 2,470 labelled photos. All the images processed were from one of the terminal gate CG02, taken throughout the year and at different times of the day. In total, there were 12 classes of signs with a very uneven distribution, following the general distribution of frequency of transporting various dangerous goods via the Skandinavienkai terminal. There are only a few or no instances of classes 1 and 7 because these items (explosives and radioactive materials) are usually not shipped via the terminal. The figure below shows the distribution of the images:



*Fig 2.2 Distributions of dangerous goods labels in the images analyzed.*

## 2.4 The Deep Learning Model – YOLO

In order to automatically identify the presence and class of dangerous goods labels on trucks, a deep learning model was trained, specifically the popular YOLO (You Only Look Once) model. The YOLO model is a computer vision algorithm first developed by Joseph Redmon et all in 2015 [[2]](http://pjreddie.com/yolo/), published in a paper titled ‘You Only Look Once: Unified, Real-Time Object Detection’. At the time it was developed, Region-Based Convolutional Neural Networks (RCNN) models were the most sought-after models for object detection (Redmon et al, 2021). The RCNN family of models, while accurate, were considered relatively slow because it employs a multi-step process of finding the proposed region for the bounding box before performing classification on these regions and then doing post-processing to refine the output.

YOLO was created with the goal to do away with multistage and perform object detection in a single stage, thereby speeding up the process. It uses a single convolutional network which simultaneously predicts multiple bounding boxes and class probabilities for those boxes.

YOLO trains on full images and directly optimizes detection performance. This unified model has several benefits over traditional methods of object detection. First, YOLO is extremely fast. Since this approach frames detection as a regression problem a complex pipeline is not needed. A new image is simply run on a new image at test time to predict detections. It can also process streaming video in real-time with minimal latency. Furthermore, at the time it was developed, YOLO achieves more than twice the mean average precision of other real-time systems. (Joseph Redmon et al, 2015).

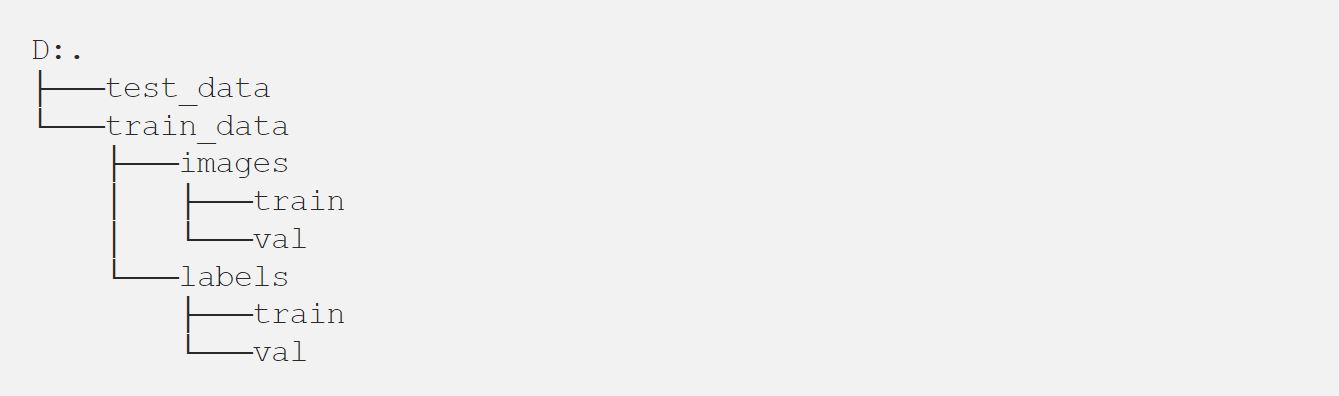
Since the release of the first version, the original authors sequentially released improved versions of the model. Following the release of YOLOv3 Redmon et al in 2018, other researchers have also contributed to the improvement of the YOLO model, culminating in YOLOv5, a Python implementation of YOLO. YOLOv5 was released in June 2020 by Ultralytics, a software development company that focuses on computer vision and deep learning.

YOLOv5 is a compound-scaled object detection model pre-trained on the popular COCO dataset and includes simple functionality for Test Time Augmentation (TTA), model ensembling, hyperparameter evolution, and export to ONNX, CoreML and TFLite. This pre-trained model can be fine-tuned on custom datasets with fewer annotated samples, making it more accessible for transfer learning and customization. It is available under version 3 of the GNU Affero General Public License [[3]](https://github.com/ultralytics/yolov5/blob/master/LICENSE) which makes it freely available for open-source projects and academic applications. For commercial projects, a paid enterprise license via Ultralytics is required.

### 2.4.1 Training the Deep Learning Model

As mentioned earlier, the pre-trained YOLOv5 model, specifically the YOLOv5m, was used for this project. The original model was customised, and by transfer-learning, the resulting model was trained on the annotated dangerous goods images.

The 2,470 annotated images were used in the proportions: 2,270 for the training session and 200 for the test session. Additional 265 background images were added to. Background images are images with no objects (i.e., dangerous goods labels). It is important to train the model on these as well, in order to reduce False Positives (FP). Ultralytics recommends about 0-10% background images. To comply with Ultralytics’ directories structure, images were organized as shown below:



The YOLOv5 Git repository was cloned and all necessary requirements were installed using the following commands:

git clone <https://github.com/ultralytics/yolov5.git>  
cd yolov5  
pip install -r requirements.txt

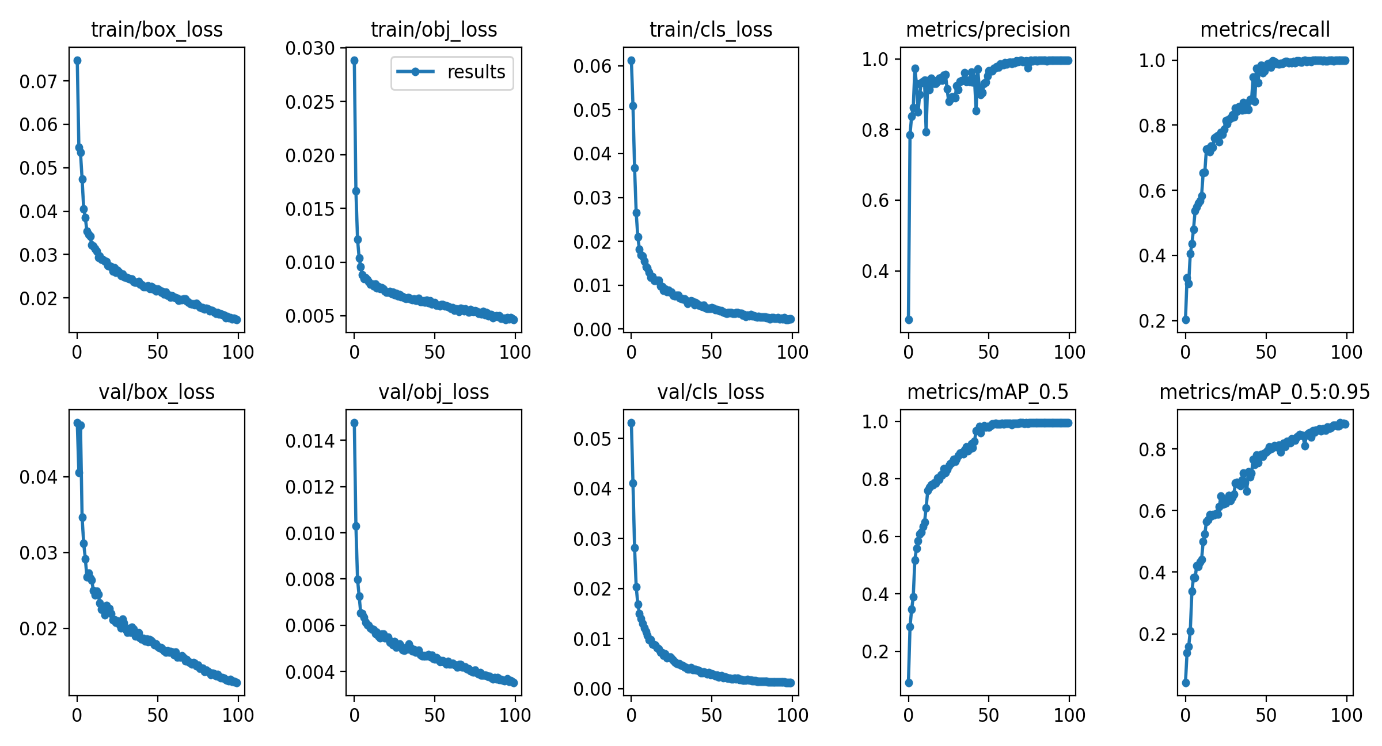
Due to high memory and CPU requirements, training was done on Google Colab. In addition, GPU computational power was leveraged by running the following command:

pip install torch==1.11.0+cu115 torchvision==0.12.0+cu115 torchaudio==0.11.0 --extra-index-url https://download.pytorch.org/whl/cu115

# 3.0 Results and Discussion

## 3.1 Training Results

After the initial setup, the model was trained on the annotated datasets with 100 epochs in 16 batches. The total run time was about 3.5 hours.



*Fig 3.1 Training Results for 100 epochs*

where:

* box\_loss: bounding box regression loss (Mean Squared Error)
* obj\_loss: the confidence of object presence is the objectness loss.
* cls\_loss: the classification loss (Cross Entropy).
* Precision: proportion of correctly predicted positive instances out of all instances predicted as positive.
* Recall: proportion of correctly predicted positive instances out of all actual positive instances

As can be seen from the group of figures above, there is continuous improvement in all major model performance metrics up to around 100 epochs. The number of epochs was subsequently increased to 200. However, this did not result in a significant increase in performance. Therefore 100 was selected as the optimal number of epochs, while considering training time and performance.

## 3.2 Test Results

To test the efficacy of the model, 200 images(different classes and background pictures) were run through the algorithm. The results are summarised in the table below:

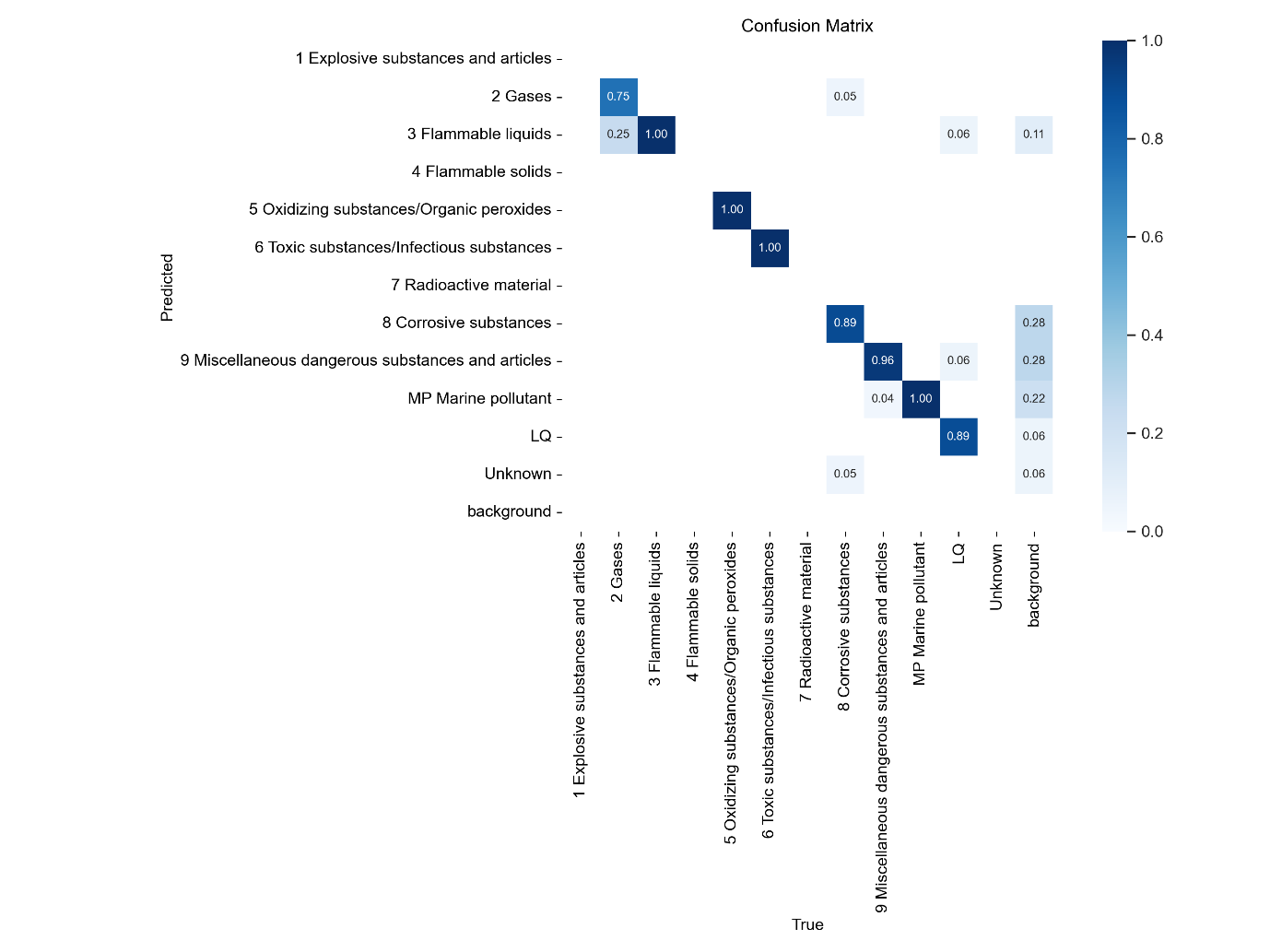
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Class** | **Instances** | **Precision** | **Recall** | **mAP50** | **mAP50-95** |
| all | 135 | 0.902 | 0.915 | 0.951 | 0.793 |
| 2 Gases | 8 | 0.807 | 0.75 | 0.781 | 0.723 |
| 3 Flammable liquids | 23 | 0.876 | 1 | 0.976 | 0.782 |
| 5 Oxidizing substances | 5 | 0.962 | 1 | 0.995 | 0.906 |
| 6 Toxic substances | 3 | 1 | 0.778 | 0.995 | 0.764 |
| 8 Corrosive substances | 19 | 0.896 | 0.905 | 0.964 | 0.771 |
| 9 Miscellaneous dangerous | 28 | 0.923 | 1 | 0.988 | 0.815 |
| MP Marine pollutant | 31 | 0.935 | 1 | 0.995 | 0.844 |
| LQ | 18 | 0.82 | 0.889 | 0.917 | 0.741 |

*Table 3.1 Test results*

To evaluate the performance of the model, some important measures were analysed. The focus is on precision and recall.

As can be seen in the table above, the model performs well with a minimum precision of 0.28 for label 2 and a precision score of 1.0 for label 6. Recall scores are also high, with the lowest being 0.78 for label 6. On the other hand, labels 3,4,9 and MP all have perfect recalls.

Furthermore, the confusion matrix below helps to better understand the results of the test:



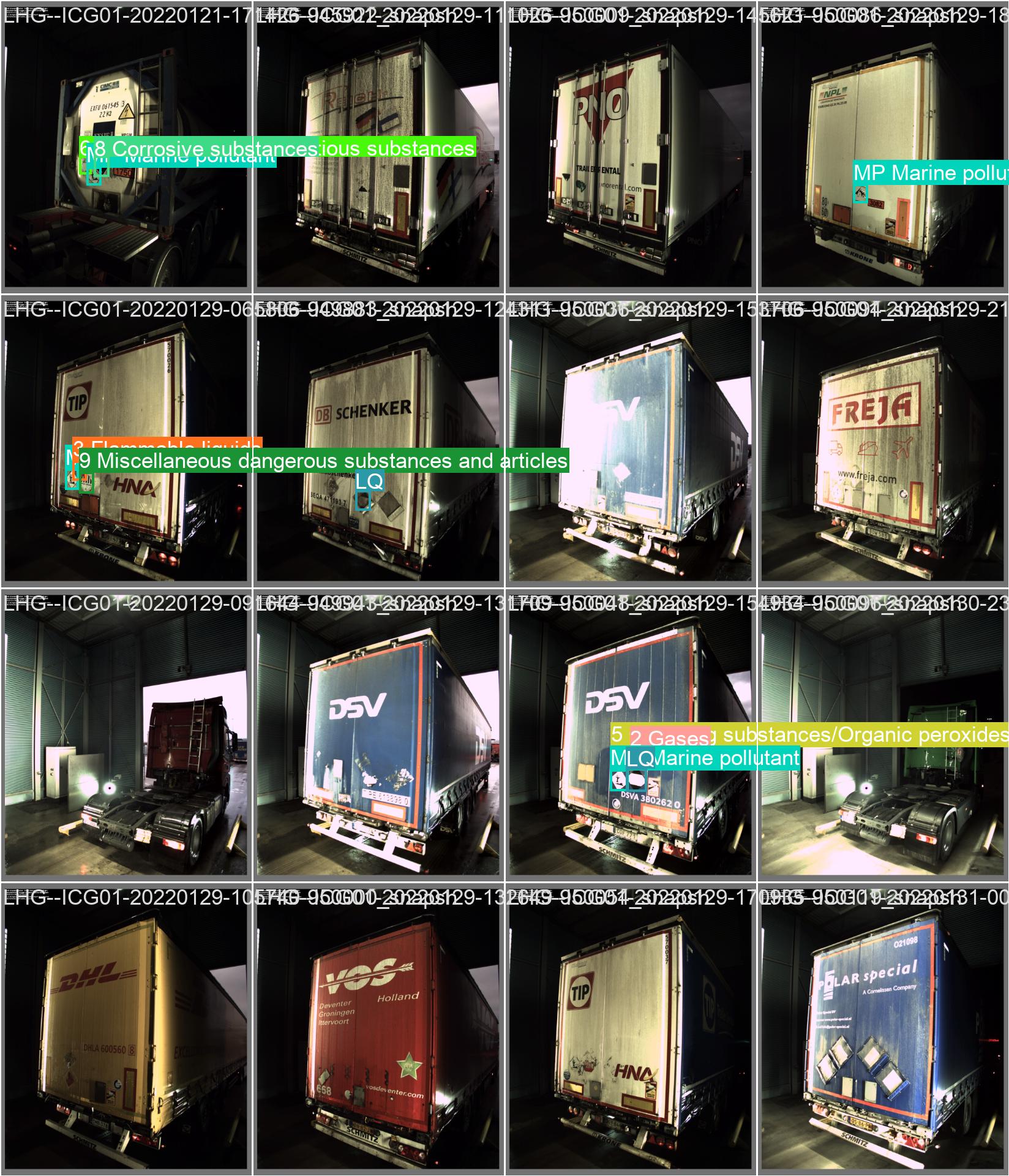
*Fig 3.2 Confusion matrix of test results*

The highest misclassifications are in label 2 where 25% of cases are misidentified as label 3. This is perhaps not very surprising as label 2 consists of three very dissimilar labels, one of which looks very similar to label 3.

 *Fig 3.3 The different types of label 2*

 *Fig 3.4 Label 3*

Below are some of the results obtained from the model test:



*Fig 2.4.4 Model test result showing the detection of dangerous goods labels on trucks*

## 3.3 Deployment

The BentoML platform was used for the deployment, and then the model was containerized as a docker image. The docker image stumprus/application\_project:yolov5m\_fh (5.92 GB) which can be downloaded and run on local machines.

By running the following command in a command line interface, the docker image can be downloaded:

docker run -it --rm -p 3000:3000 stumprus/application\_project:yolov5m\_fh serve

Subsequently, the model call be accessed in a web browser via the local web address:

http://127.0.0.1:3000

## 3.4 Recommendations and next steps:

The recommendations below are based on the outcome of the project:

* Adopt this model for identifying dangerous goods labels on trucks.
* In the meantime, perform second-level checks when label 2 is detected. This is due to relatively low recall for that class and its misidentification as label 3.
* Retrain the model, but split label 2 into its constituent sub-labels before training
* Integrate the model with the existing booking data from the LHG logistics system for auto verification

# 4.0 Conclusion

This project was aimed at automating the detection of dangerous goods labels and their respective classes on trucks at the Skandinavienkai RoRo terminal of Lübecker Hafen-Gesellschaft mbH (LHG). To do this, a deep learning model Convolution Neural Network (CNN) model, Yolov5 was trained on the images of trucks obtained from LHG. The resulting model performed very well with generally high precision and accuracy across most classes except label 2 which bears striking resemblance with and is sometimes misidentified as label 3.

With the current performance, the model can be adopted to move the current process from manual to semi-automated, while improvements are made to make it fully automated.

Importantly, the outcome of this project shows that the full automation of the scanning and verification system at the Skandinavienkai terminal can be achieved, and indeed, has been achieved with the resulting object detection model.

This will lead to better efficiency and effectiveness of the operations at the port and eliminate potential human errors from the current manual process.

# 5.0 References

UN Recommendations on the Transport of Dangerous Goods

<https://unece.org/fileadmin/DAM/trans/danger/publi/unrec/rev21/ST-SG-AC10-1r21e_Vol1_WEB.pdf>

You Only Look Once: Unified, Real-Time Object Detection Joseph Redmon∗ , Santosh Divvala∗†, Ross Girshick¶ , Ali Farhadi∗† University of Washington∗ , Allen Institute for AI† , Facebook AI Research¶ <http://pjreddie.com/yolo/>

YOLOv5 LICENSE:

<https://github.com/ultralytics/yolov5/blob/master/LICENSE>

A Brief History of YOLO Object Detection Models From YOLOv1 to YOLOv5 By Gaurav Maindola -August 27, 2021

<https://machinelearningknowledge.ai/a-brief-history-of-yolo-object-detection-models/?utm_content=cmp-true>