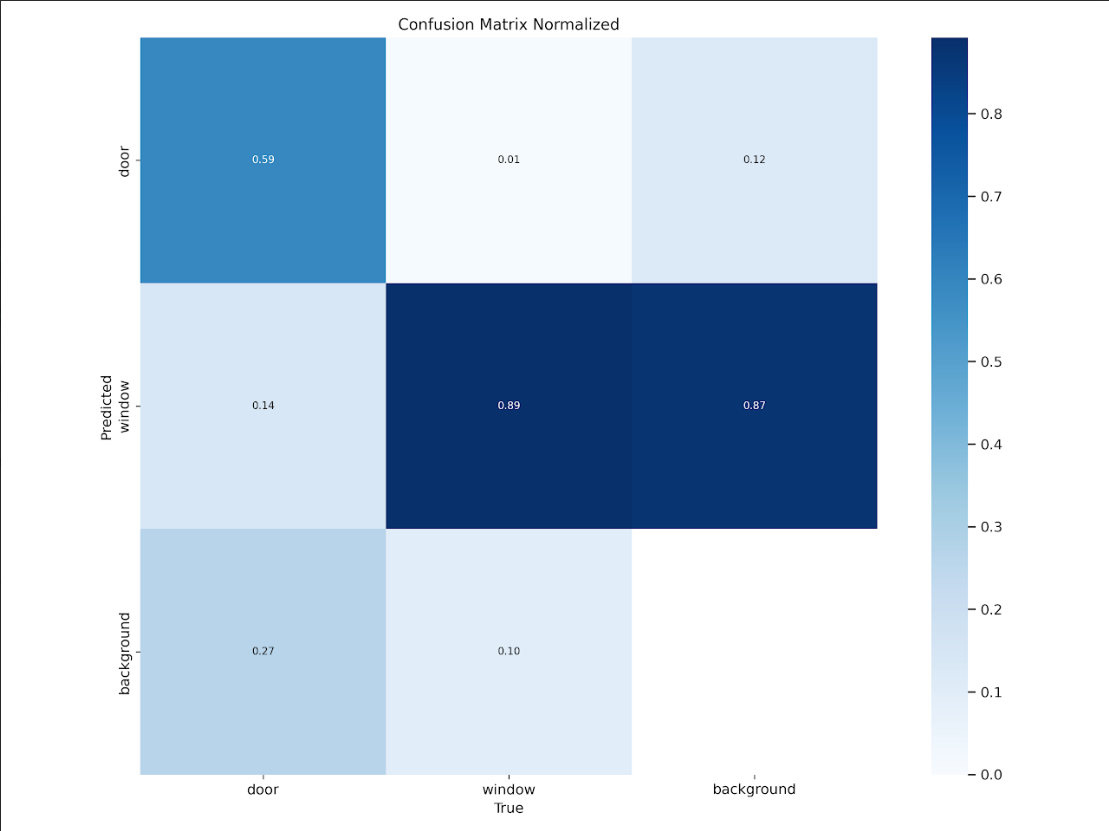
**Report**

We have a model trained on the Turkey dataset. After 20 epochs it seems to not increase the evaluation results. It seems like it is good for windows, but there is still room for improvement for doors.



The model when running an image from the KW dataset.



Observations:

* after 2 simulations I ran out of GPU
* If we increase the batch size to 32 images we run out of VRAM
* When running the notebook, use the scripts to save on the drive the results of the training and validation phase and also save the weights of the model. Be careful with the google drive paths so that you don t override the results from somebody else

After looking over your links, we found this model that does object detection for multiple classes (<https://universe.roboflow.com/demo-mt7uu/house-models-yq1pt/model/7> )   


Then there is this model for image segmentation (<https://universe.roboflow.com/blackhouse/yolo_sam_auto_annotations-kxeir/model/4> )



The task of object detection is defined for us, whereas what image segmentation consists of is to draw these boundaries at the pixel level. Overall, it is considered to be a more difficult task. In terms of datasets, there seems to be in roboflow some for this task, but already what we have for object detection from Turkey represents more data than the stuff for image segmentation. I suggest to continue working on object detection, since one of the tasks in the end is the ability to count the objects, which is sufficient to be done using object detection.

In our first instance, we have to detect

* Windows
* Doors
* Balcony
* Building

Tasks

* General task (read this research paper so that we are all more or less familiar with what we need to do. It is quite short and if there are questions about metrics, just ask GPT. (<https://drive.google.com/drive/folders/11v7-sUhnFuNzg_NAdkhsDt3xG2023-w8> )
* Explore more the business context of our model for the current tasks ( the 4 object classes) 1 person
* Create bounding boxes for 50 images from the KW dataset.(tutorial: <https://www.youtube.com/watch?v=pJaM06FG-wQ&ab_channel=Roboflow>) Choose the images that go perpendicular on the building and try to use different buildings (look over the megafile: (<https://mega.nz/folder/QfF10ATY#xQnM-u67i4MrQ3DMefUQLw/folder/oOkh3b4B> ) and also in Spotr (<https://docs.google.com/document/d/1WAsHniatqRs15RY6vNERg5zYrcPKR-N-FEXm6JbCQbo/edit?tab=t.0> ) and create the bounding boxes for windows, doors, balconies and building. (1 person) (Florentijn: I can do this on Tuesday)
* Find 2 more research papers and their datasets that are relevant to our case study and make a summary of the main points like methodology (1 person) (Yan : I can do this by monday night) Pro’s & Con’s of object detection and segmentation
* Find more relevant datasets for our main objects. Be careful to be bounding boxes for object detection and not image segmentation. See the labels to be YOLO format or something that later can be parsed easily to YOLO format. The format that we want is:  
   **1 0.29447115384615385 0.40625 0.1875 0.109375**  (class\_id x\_center y\_center width height ) and the format for image segmentation looks like this   
   **11 0.17902734375 0.4785107421875 0.1976181640625 0.3754599609375 0.143375 0.36824218750000004 0.1252939453125 0.472908203125 0.17902734375 0.4785107421875** ( class\_id (x1, y1, x2, y2, ..., xn, yn) represents the normalized coordinates of the polygon that outlines the object's shape.) From what I saw, sometimes we can reuse for windows some datasets for image segmentation, since they just draw polygons over them, but in the end I think we still need to select them manually and eliminate the other classes that are not corresponding to our task. (Friso)
* Modelling and interpreting the results (2 people) (Eugen)

## Pro’s and Cons of object detection and image segmentation

**Pro’s of object detection:**

Object detection models (e.g., YOLO, SSD, Faster R-CNN) are often faster to train and run because they only need to locate objects and define them with bounding boxes.

Bounding box outputs are easier to interpret and process for downstream tasks, such as counting or spatial analysis.

Works well for straightforward identification tasks when precise boundaries are not critical.

**Cons of object detection:**

Bounding boxes may include extra background or exclude parts of objects, making them less precise for tasks requiring exact object boundaries.

It does not capture the shape or outline of the objects, which may be essential for architectural analysis or complex layouts.

Struggles when objects overlap significantly, as the bounding box representation may fail to separate them clearly.

**Pro’s of segmentation:**

Enables extraction of detailed shapes and boundaries of architectural elements, which is important for CAD or 3D modeling.

Can distinguish between overlapping or closely placed objects by assigning unique pixel regions to each.

More suited for nuanced tasks, like identifying cracks in windows or decorative patterns on doors.

**Cons of segmentation:**

Requires more memory, processing power, and training time than object detection.

Creating pixel-level ground truth data for training can be labor-intensive.

## Extra papers for our case

Zhang T, Li J, Jiang Y, Zeng M, Pang M. Position Detection of Doors and Windows Based on

DSPP-YOLO. *Applied Sciences*. 2022; 12(21):10770. <https://doi.org/10.3390/app122110770>

Chatgpt with search summary:

The paper titled "Position Detection of Doors and Windows Based on DSPP-YOLO" presents an enhanced algorithm, DSPP-YOLO, designed to improve the accuracy of detecting doors and windows in complex, unknown environments. This enhancement is achieved by integrating DenseNet and Spatial Pyramid Pooling (SPP) into the YOLOv3 framework, allowing for better feature extraction and multi-scale receptive field fusion. Additionally, the authors employ the K-means++ algorithm to re-cluster candidate box sizes, reducing errors associated with candidate boxes. Experimental results indicate that DSPP-YOLO achieves a detection accuracy of 77.4% for doors and 38.1% for windows, marking improvements of 3.3% and 8.8%, respectively, over the standard YOLOv3 algorithm.

Note: the kmeans algo to recluster might be interesting for us

Nordmark, N., & Ayenew, M. (2021). Window detection in facade imagery: a deep learning approach using mask R-CNN. *arXiv preprint arXiv:2107.10006*. <https://arxiv.org/pdf/2107.10006>

Chatgpt with search summary:

The paper "Window Detection In Facade Imagery: A Deep Learning Approach Using Mask R-CNN" by Nils Nordmark and Mola Ayenew explores the application of the Mask R-CNN framework for detecting windows in building facade images. This task is essential for urban analysis, semantic reconstruction, and digital twin creation. The authors employed transfer learning, initializing their model with COCO weights, and trained it on a custom dataset of street-view facade images. Their approach achieved instance segmentation of windows, delivering results comparable to previous state-of-the-art methods, even without additional post-optimization techniques.