Image Analysis Internship - database inspection

Nicolas Le Roux

September 10, 2023

1 Introduction

This is a summary of work conducted by Nicolas Le Roux, a former intern from ENSTA Bretagne, France, over a matter of weeks in summer 2023. The purpose of this work was to evaluate the default panoptic segmentation model (panoptic_fpn_R_101_3x) that comes with Meta's Detectron2 module for Python. To do so, a database of 5100 images obtained from Twitter was used. The images were gathered from 2021 Belgian activity on the platform, and many of them are reactions to the manhunt for Jürgen Conings. Each image was analyzed by the model, and an annotated image was returned, with masks and labels for each class of the panoptic segmentation. These images were then manually inspected, to determine the model's strengths and weaknesses. Furthermore, two different sets of 1000 and 90 images were inspected more closely, with each *thing* (cf. COCO) class prediction individually assessed for veracity. These two were used for a more statistical analysis of the model.

Remarks:

- Throughout this document, the mentions *thing* and *stuff* shall refer to their classification according to COCO.
- In all the following images, the left half is the original image, and the right is the annotated image with masks provided by the model. Also, in the top-left corner of the annotated image, the percentage of unidentified pixels is given. Creating the annotated image takes up most of the run-time when compared to simply detecting, locating, and returning the different elements. This step is, of course, optional.
- Throughout this document, art shall be defined as a representation of something: *things* and *stuff* (cf. COCO), decorative or otherwise cosmetic additions to a digital format (like on a website or presentation), or anything belonging to a conventional art form.
- All the code and images used for this work are publicly available on Github.

2 Partial statistical analysis of the database

Before inspecting the specifics of the model, in this section the focus will be on a simple statistical analysis of the annotated database. Indeed, presenting the strengths and weaknesses of the model

might induce unfair bias in a reader unless they know the likelihood of these occurrences. The aim of this section is to determine a confidence score threshold, a threshold that a human operator can trust to give good results according to their needs, as well as determine the ideal size for images. As will be seen bellow, the model is not perfect, and it might be in the interest of an operator to be more discerning than the model. As mentioned above, two different sets of images, with 1000 and 90 images respectively were inspected more closely. For each, a note was generated using the analysis data, containing for each image a list of all detected *things* and their associated confidence scores. The 90 images one also had the *stuff* classes. These notes were then completed with manually-typed labels for each prediction. The labels were:

- F, meaning the prediction was incorrect and made on a real thing (not artwork, cf. bellow).
- C, meaning the prediction was correct but on artwork.
- I, meaning the prediction was incorrect and on artwork.
- " " (an empty space), meaning the prediction was correct and made on a real thing (not artwork).

For the purpose of an easier reading of this document and its graphs, the " " were replaced by "T" in the explanations and graphs. They did not bear this label in the notes in order to hasten its completion, since most mentions were " "/"T".

2.1 General performance

This subsection deals with the results of the 1000 images note. The numbers of each label, as well as their associated confidence score, were tallied and used to produce the graphs bellow.

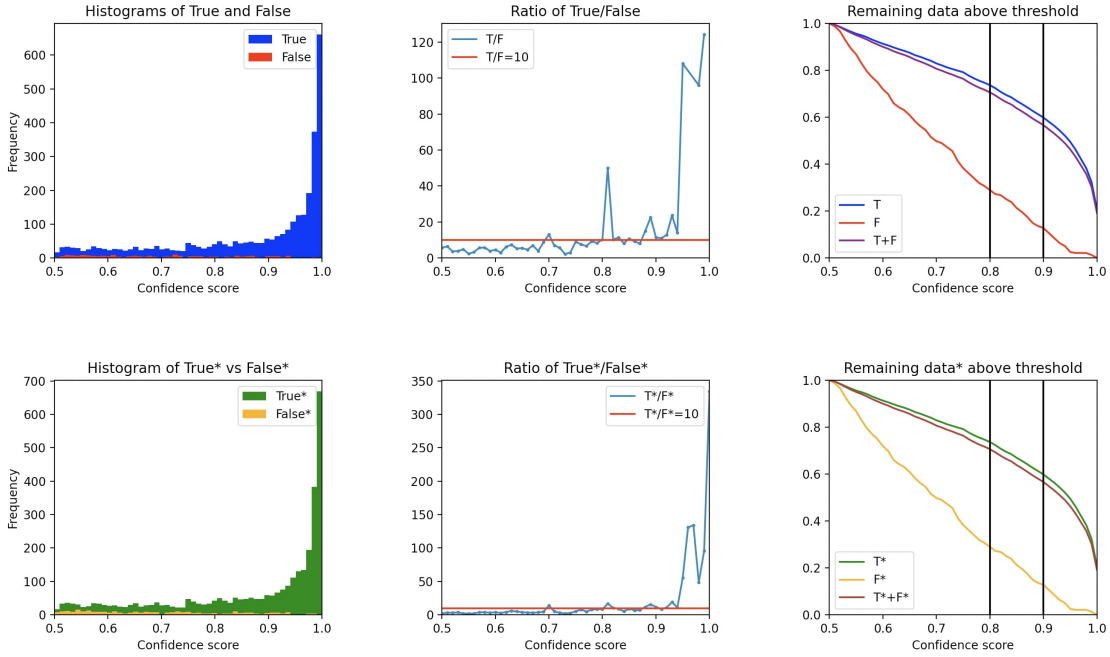


Figure 1: Statistics on the database

The first row of graphs tallies only T and F mentions, meaning only data taken from real, photographic *things*. This distinction was made to incorporate only data comparable to the training sets, which did not include artwork (at least, not to any useful degree). The second row tallies all four mentions. The confidence score values range from 0.5 to 1, in increments of 0.01, and ordinate values are present for any valid abscissa.
The first column contains histograms of the general result populations:

- The top one compares only the histogram of T and that of F. It is clear that there are far more T labels than F ones, especially on the end of the spectrum.
- For the bottom one, the labels T* and F* were created. T* corresponds to both T and C labels, F* to F and I labels. Here, one histogram contains T* labels, and another contains F* labels, and the graph looks rather similar to the top one. The proportions are slightly different, but it seems that the ratio of valid identifications over invalid ones follows a similar trend to the first graph: it is proportional to the confidence score.

The second column contains graphs of the ratios of the aforementioned categories. In both graphs, a line of a 10x ratio was plotted as well, as a possible, but arbitrary, threshold for a useful model.

- The top only deals with T over F. Here, valid abscissa, ones with a corresponding ordinate, are ones where both T and F values exist at the corresponding confidence score. For any valid abscissa, the ordinate corresponds to the ratio of the histogram value of T over the histogram value of F at that particular confidence score. It represents how many more T labels can be found than F ones at that score. As can be seen on the graph, the values are vaguely proportional to the confidence score. And above 80%, nearly all ordinate values are above the line.
- The bottom one deals with T* over F*, and works the same way as the top graph. It is interesting to note that these graphs reveal greater differences between the T/F and T*/F* sets than the previous graphs. Still, like the top one, the values eventually exceed the 10x threshold, at around 90%.

Finally, the last column of graphs represent the populations of scores. At a given abscissa, there are three values corresponding to three different curves. For each curve, each value corresponds to the percentage of the population (for that category) remaining at an equal or higher confidence score. The point is to decide on a score threshold that still retains a significant portion of the available data: there is no use in having a 99.99% certainty if 99% of data is filtered out. In both graphs, two potential score thresholds were drawn, to help with visualization.

- In the first graph: at 80% threshold, 74% of T values are still there, as well as 70% of all T+F values, and only 29% of errors are left (n.b. at this point, T values are 10x more common than F ones). As for the 90% threshold, we have 60% of T, 57% of T+F, and 13% of F left.
- In the second graph, the percentages are the same, but corresponding to T*, T*+F*, and F*.

2.2 Influence of image size

In order to best use the model, an optimal size for images must be determined. One that balances low memory usage, short analysis time, and a comprehensive description of the image's contents. To that end, a small set of 90 images were chosen from the 5100, none of them included *art*. They were reshaped into squares with sides of various sizes: 800px, 400px, 200px, 100px, and 50px. The first size, 800px, was chosen since it is the closest hundreds value to many of the larger images present in the original database. All subsequent ones are halves of the previous values. Each of these new 450 images were analyzed and their predictions inspected manually, leading to the following graphs. The *stuff* classes were ignored when making the graphs, since they are harder to judge.

(a) 800px (b) 200px (c) 50px

Figure 2: A same image at different sizes

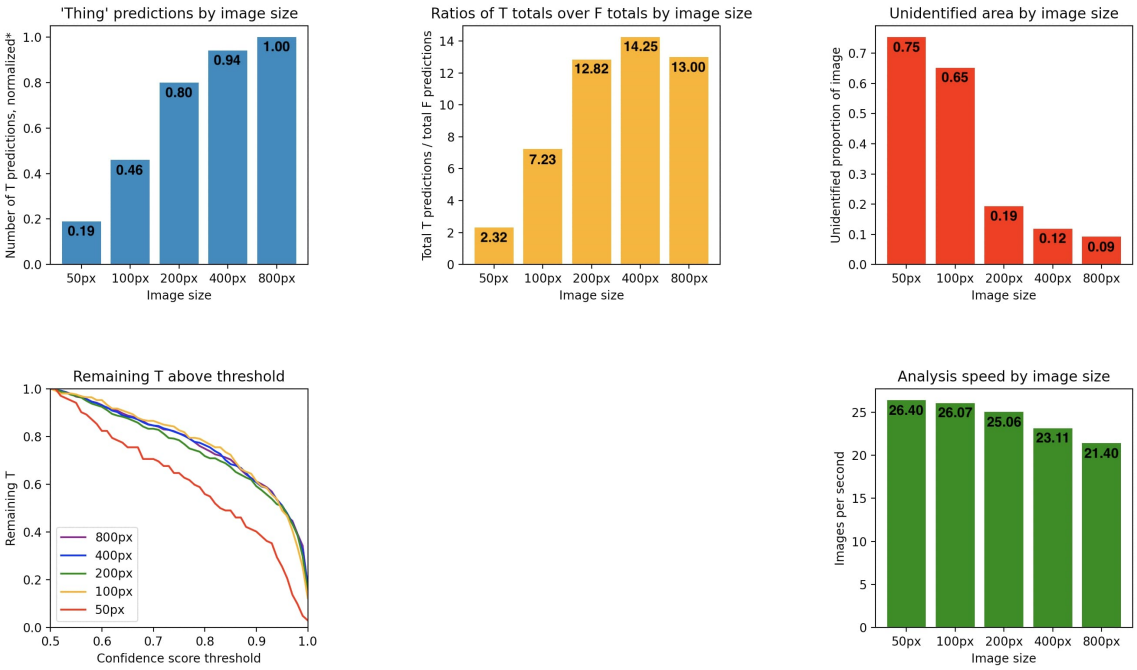


Figure 3: Influence of image size on model performance

- The first graph presents the total number of correct (T) predictions made by the model on the same set of images (albeit at different sizes). The results were normalized using the values from 800px, the largest size candidate. Its purpose is to determine the influence of image size on the quantity of valid information received.
- The second graph serves to represent the proportions of T and F predictions made by the model at different sizes. At each size, the sum of all Ts was divided by the sum of all Fs. These do not include data on any confidence thresholds, since there were too few errors to properly represent a graph from 50% to 100%. Its purpose is to determine the likelihood of valid information over invalid ones.
- The third graph deals with the unidentified parts of the image, that which the model declines to classify. If a model cannot identify most of an image, then perhaps there is something wrong with the image. This graph's purpose is to determine the proportion of wasted information in an image at a given size.
- The fourth graph is similar to one seen earlier in this document, it represents, at a given confidence score, the amount of correct (T) predictions left at or above that threshold. Its purpose is to determine if a confidence threshold's worth depends on image size.
- The fifth graph is about time. When running the image analyses, the time needed to perform the task can be of vital importance, depending on the mission. To obtain this graph, 60 out of the 120 possible permutations of [800px, 400px, 200px, 100px, 50px] were chosen, and the analyses ran for the image sizes in those 60 different orders. The reason for alternating the orders is because it seemed like the first analysis was consistently slower than subsequent ones in a given compilation. This graph's purpose is to inform on the influence of image size on analysis duration.

Conclusion: In the end, a trend appears in the graphs, it seems that the three largest sizes all have similar performances. There is a clear degradation in performance when jumping from a 200 pixel square to 100 pixels, be it in number of predictions, ratios of T over F, or unidentified area. Furthermore, the advantage of smaller images in run-time is too negligible to justify their use. Therefore, it would be advisable to opt for sizes greater than or equal to 200px. As for *stuff* classes, they became rarer as image size shrunk, and the 50px ones contain no *stuff* identifications. It is also worth noting

that *person* identifications are among the last, mostly, correct ones at lower sizes, proof that the model was heavily trained on photographs of humans.

2.3 General conclusion

In the end, the decision to implement a threshold and a standard image size will be different depending on operational requirements. Must one prioritize minimizing the proportion of errors? Or the volume of valid data? Based on the data available, a confidence threshold of 80% and an image size of 200 pixels per side seem reasonable.

3 What the model gets right

After inspecting thousands of images, a resounding impression was formed: Detectron2's default panoptic segmentation model is very accurate. It is a valuable tool for quick analysis of images, and vastly exceeds the human ability to analyze a scene and describe it. The data shown previously testifies to this. Still, those are just numbers, and this is a visual tool, therefore a visual presentation is in order, along with a detailed description of the model's strengths. The following subsections are in descending order of how impressively the model in each.

3.1 *Stuff*

The model's greatest strength, arguably, lies in its very accurate detection and identification of different *stuff* classes. It performs admirably in a wide range of natural, urban, and lighting settings:

- Natural elements, like the sky, trees, grass, and others, are generally well segmented.

(a) Forest (b) Riverside

Figure 4: Model is well-trained on natural settings

- Lower luminosity does not greatly diminish the model's capabilities, assuming there is some amount of light. The boundaries may not be exact at night, but its ability to detect elements is impressive.

(a) Dusk (b) Nighttime

Figure 5: Impressive performances in lower visibility

- Similarly, urban settings are well identified. The exact boundaries might be imperfect at times, but the different possible classes tend to show up.
- All kinds of construction materials are identified easily, and even if the boundaries aren't perfect, the descriptors that the model can give are very informative.

(a) Train station (b) Building facade

Figure 6: Urban settings

- Interior design elements are also skillfully recognized, albeit with imperfect delimitations. This includes four different types of specific wall materials, mirrors, tables, etc.

Conclusion: The model is very capable when it comes to detecting *stuff* classes in an image. Whilst its ability to precisely draw masks around those classes can, sometimes, falter, it is generally competent with a wide range of materials. For the purpose of obtaining a description of the image background, it is highly effective.

3.2 *Things*

The *things* category is perhaps the most important when analyzing an image, since oftentimes the subject of a photograph is a *thing*, as opposed to *stuff*. There is not much to say about this category, since confidence is a major parameter that can give insight into the reliability of a prediction. Thus, this subsection will focus mostly on some general features of detection, and conditions that the model can handle with relative ease:

- If a large number of instances of a *thing* are present in the foreground, then the model can easily separate the individual members, and detect instances that are partially covered. Using this model could allow for a quick, approximate count of a large crowd.

(a) Crowd 1 (b) Crowd 2

Figure 7: Crowd counting

- Blurring/pixelating faces does not hinder detection of persons, much like partial cover does not either.
- Wearing a costume or heavily-concealing outfit does not seem to hinder detection. This could be due to chance with training data, but the model's ability to recognize people is impressive.

(a) Abstract costume (b) Burqas

Figure 8: Costumes don't conceal you

- Persons in black-and-white photography are also detected, so color is not necessary for at least one *thing* class.
- Fencing does not prevent detection of objects.

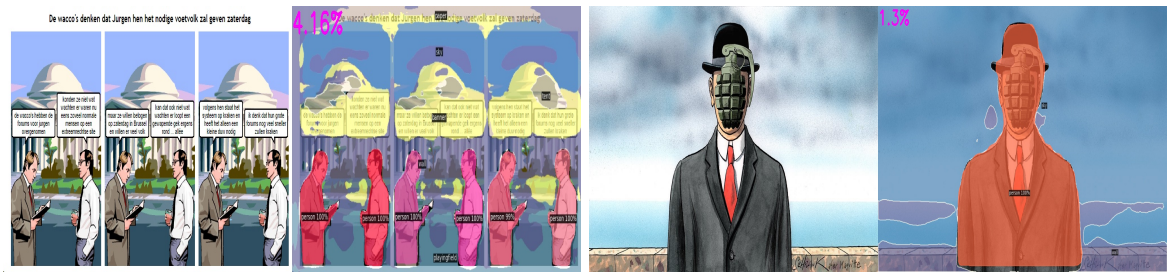
Figure 9: Wire fence does not conceal things

Conclusion: The panoptic segmentation model has likely been trained on a wide variety of different versions of its classes. This is at least true for persons, which is helpful since this project is likely to focus on content involving people.

3.3 Digital and physical art

The final category to examine in this subsection is one that the model was not trained on. Nonetheless, there are some inconsistent instances of the model being correct in its predictions, even very confident in them:

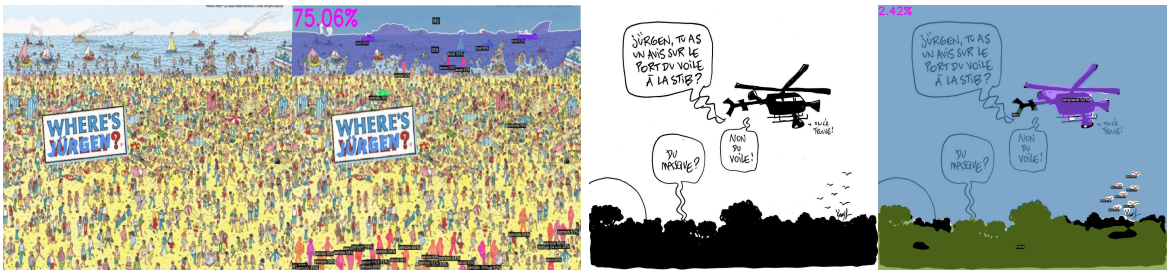
- Certain styles of drawing yield confident, correct results, especially when it comes to people. It seems likely that the *person* class has received special training. Unfortunately the model cannot tell a user if a detected person is real or not.



(a) Cartoon people (b) Cartoon person

Figure 10: Certain cartoon styles can be understood

- Some other elements have been correctly identified multiple times: birds drawn like marks, helicopters (although called airplane due to COCO limitations), and even clouds.



(a) Cartoon beach (b) Cartoon sky

Figure 11: More cartoon elements correctly identified

Conclusion: Under some circumstances, the model is capable of recognizing some elements of a cartoon/drawing. However, any success obtained with cartoon images is highly inconsistent and the model should not be characterized as capable of analyzing cartoons. Its specific flaws on the matter will be explored later.

3.4 General conclusion:

Detectron2's default panoptic segmentation model has many strengths: from delimiting natural elements, to detecting partially-obscured objects, to operating in a variety of conditions, and much more. In the right hands, it could be an extremely valuable tool for data extraction.

4 What the model gets wrong

Now that the model's many strengths have been laid bare, it is important to also address its weaknesses. Whilst they are statistically rarer, they do occur, and ignoring them would be irresponsible. Bellow, the same categories as above can be found, in rough descending order of gravity of errors.

4.1 Digital and physical art

Overall, the model's ability to correctly identify *things* and *stuff* in art is indeed inconsistent. It is clear that the model was not trained on art, and therefore any positive results are more the lucky exception than the rule. In this section, the negative aspects of that inconsistency will be outlined.

Here are some characteristics of the model:

- The model tends to draw arbitrary delimitations in uniform backgrounds.
- It can sometimes be quite confident in its wrong guesses.

(a) Arbitrarily delimited sky (b) Aarakocra (birdman) is person and horse

Figure 12: Delimitation and confident errors

- It tends to hallucinate things in camouflage patterns, with often confusing delimitations.
- Presence of white background is often interpreted as paper, which could be used as a benchmark.

(a) Random *things* and *stuff* in simple camouflage (b) Online article is paper

Figure 13: Camouflage hallucination and white backgrounds

- It often confuses emojis for sports balls, which is interesting because the model seems bad at recognizing actual sports balls.

(a) ex 1 (b) ex 2

Figure 14: Emojis are easily, but consistently misread

Furthermore:

- It does not understand maps (Google, traditional, etc.), being unable to return anything for a human operator to recognize a map.
- Color gradients can greatly impact recognition, causing hallucinations of delimitations, and making simple art sound like a complex scene.

Conclusion: Having artwork mixed in with real world data is dangerous and will lead to an increased rate of false positives. Finding a way to separate artwork from photographs would be of tremendous use in limiting the damage. Perhaps the proportion of "paper" in an image could be used as a benchmark?

4.2 *Things*

In this second subsection, *art*, as defined above, shall be ignored, and the subject will be the model's failings in regards to *things* (cf. COCO). These are classes of countable objects, unlike *stuff*, like persons, dogs, cars, windows, etc.

A first group of common inaccuracies is that of consistently misidentified *things*, objects that have no relevant class with the COCO database:

- Helicopters are consistently described as *airplanes*, with high confidence scores. This could lead to confusion, when reading the list of detected classes, about the specific aircraft present.
- Military terrestrial vehicles not easily compared to civilian ones, like tanks, are often identified as *trucks*. None of the default COCO classes are unique to the military, so identifying a scene of military activity using the model might be tricky.

(a) Tanks misidentified as trucks

(b) Helicopters misidentified as airplanes

Figure 15: Consistently misidentified vehicles

Now for some less organized by still common errors:

- The model is not great at distinguishing between different animals, and will make guesses at animals that are not within its database. As long as the model is not used for zoological studies, perhaps any animal detection could be rewritten to say "animal", instead of a specific species, thereby limiting inaccuracies.

(a) Snakes are not birds

(b) Deers and foxes are not dogs

Figure 16: Detectron2's failures in zoology

- Similarly, it is not adept at distinguishing between sometimes similar-looking variants of a class: like similar-looking fruits, or road signs.

(a) Everything's an apple

(b) Not exactly a stop sign

Figure 17: More not-quite-correct guesses

Overall, it seems that if an object has characteristics belonging to an existing class, the model is comfortable with making approximations. In many cases, as discussed above, a workaround would be to simplify the descriptions to more broad terms. However, in cases where an object is too different from any existing class, the model does not hesitate to leave them blank. This is most often the case with firearms and weapons of war. An amusing note: clips have been misidentified as *handbags* on multiple occasions. This is however not consistent enough to offer a workaround.

(a) Rifles are not recognized

(b) Bazookas are not recognized either

Figure 18: Detectron2 is made for love, not war

Finally, some other notes that might not need a figure:

- If an object is partially covered by another, resulting in a *thing* being visually split in multiple parts, then the model is likely to count each part as separate instances of the same class. This is a problem if the exact count of a class is important.
- Objects in the foreground, that are therefore bigger, are more easily detected than objects in the background.
- A zombie-like character from a movie was misidentified as a *person*.
- *TV*s are a common interpretation for text on a natural background, or for black lamp shades at the right angle.
- Anything vaguely circular tends to be called a *frisbee*.
- All kinds of tables are labelled *dining tables*.
- *Kites* are often just flags.
- In large crowds with people in both the foreground and background, background persons are often missed if small enough, or given a lower confidence score.
- Blurring a person's face has little effect on identification as a person.
- Small boats have been misidentified as *birds*

- etc...

Conclusion: Most of the model's problematic mistakes have to do with the *thing* classes, with plenty of misleading identifications. This is a problem if one wishes to obtain a description of the image using Detectron2's default panoptic segmentation model. The solution to these issues is three-fold:

- Implement a threshold for filtering out lower confidence scores. This should eliminate most inconsistent, false identifications.
- Generalize certain keywords to be part of a broader whole: animals could be *animal*, fruits and vegetables could be *produce*, aerial vehicles could be *aircraft*, *stop sign* replaced by *road sign*, etc.
- Interpret certain classes as likely to be something else: trucks might be any big terrestrial vehicle, banners/curtains could be flags in the right context, anything circular could be called frisbees and sports balls.

4.3 *Stuff*

In this final subsection the focus will be on *stuff*. This term refers to uncountable classes of objects, like dirt, sky, water, etc. Although the model has demonstrated its strengths, especially when it comes to identifying *stuff*, it is not without fault. Bellow are some of the more common errors it is prone to:

- The model struggles with complex or ambiguous boundaries between certain classes, like: road and pavement, interior materials, some nature elements, etc. Although it generally does not hallucinate elements that aren't there, their exact boundaries might be inexact.

Figure 19: Awkwardly segmented road and yard

Figure 20: Seemingly confusing windows and interior design

- Smoke causes the model to hallucinate random classes, and leads to perplexing results for the image analysis.

Figure 21: Terribly confusing smoke

- Blurriness hinders identification, increasing the chance of hallucinations.
- What exactly constitutes a *mountain* (cf. COCO) is somewhat vague, sometimes a broken treeline is enough to qualify as such. This can give a wrong impression of terrain and location.

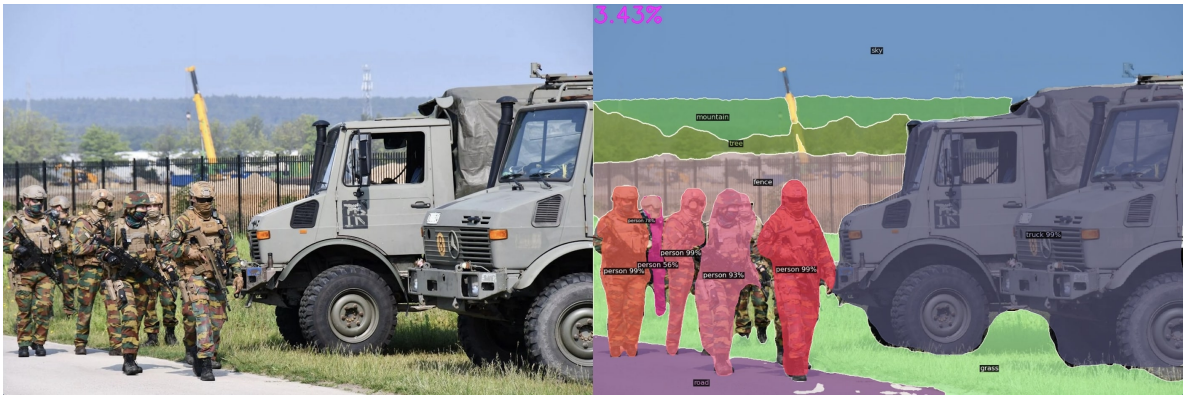


Figure 22: World's flattest mountains

- Flags are not a recognized class by the model, and often classified as banners, curtains, or other. This is bothersome since their identification is inconsistent, allowing no simple workaround.

Figure 23: What is a flag?

- Satellite or aerial photography is poorly handled, to a degree far inferior to that of a human operator.

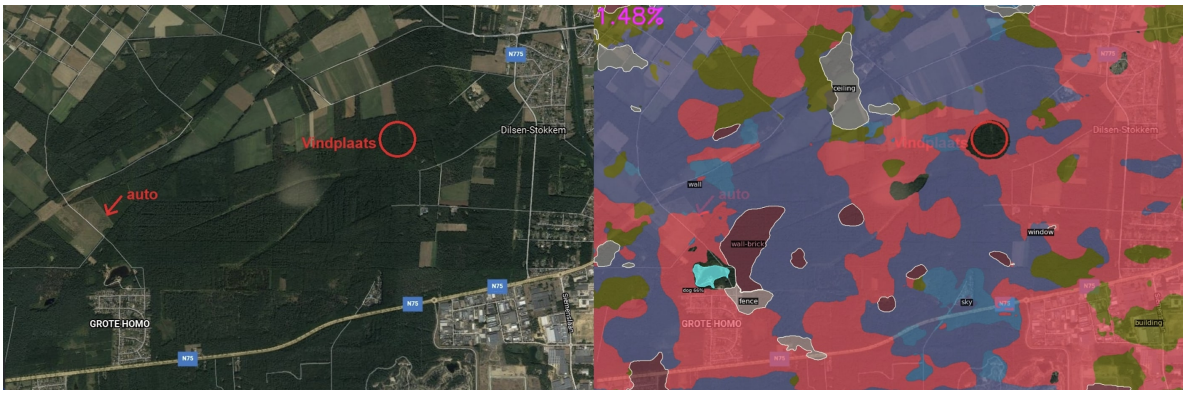


Figure 24: Satellite view is a jumbled mess

Conclusion: The model has some unique flaws when it comes to *stuff* classes, the most common ones can be worked around by knowing how to interpret results:

- *mountain* can mean hills, a break in a treeline, or the real deal. They are all nature, perhaps that is enough.
- Banners or curtains with a large amount of persons, or surfboards without any water in sight, are a likely sign of flags at a protest.
- Dealing with smoke could be done by filtering out small (compared to a predetermined threshold) *stuff* classes.

In the end, the most common weakness of this model when it comes *stuff*, is properly drawing boundaries between classes. If a human operator is not interested in the locations of everything in an image, just their presence, then this is a non-issue.

4.4 General conclusion

When laid out like above, Detectron2's default panoptic segmentation model might seem too unreliable for most classes and misleading in a number of cases. However, it is important to remember that in its intended use cases: not artwork, it is statistically far more likely to be correct than incorrect. Still, error mitigation is a necessary step towards ensuring a more reliable analysis. In the previous, lesser conclusions, a variety of possible paths towards mitigating errors were evoked. If one were to desire the most general way to reduce the occurrence of false classification, it would be to introduce two thresholds: a confidence threshold for *things* and an area threshold for *stuff*. This is in line with the conclusion of the first section, by statistical analysis, and by the impression that the manual inspection of the database left on the author.

5 Ultimate conclusion

All in all, Detectron2's default panoptic segmentation model proved to be a useful tool with great potential for analyzing a wide variety of photographs. However, alone, it is too inconsistent to be trusted with anything requiring a higher degree of confidence. Fortunately, with a few additions, like a confidence score filter, an area filter, an art filter (which remains to be devised), and others mentioned above, its reliability should increase greatly. It is well suited for images of all kinds of sizes, but somewhere between 200 and 100 pixels a side, it starts to crumble, with smaller images being impractical. As for possible future studies, another important factor to study would be the ease of training of the model.