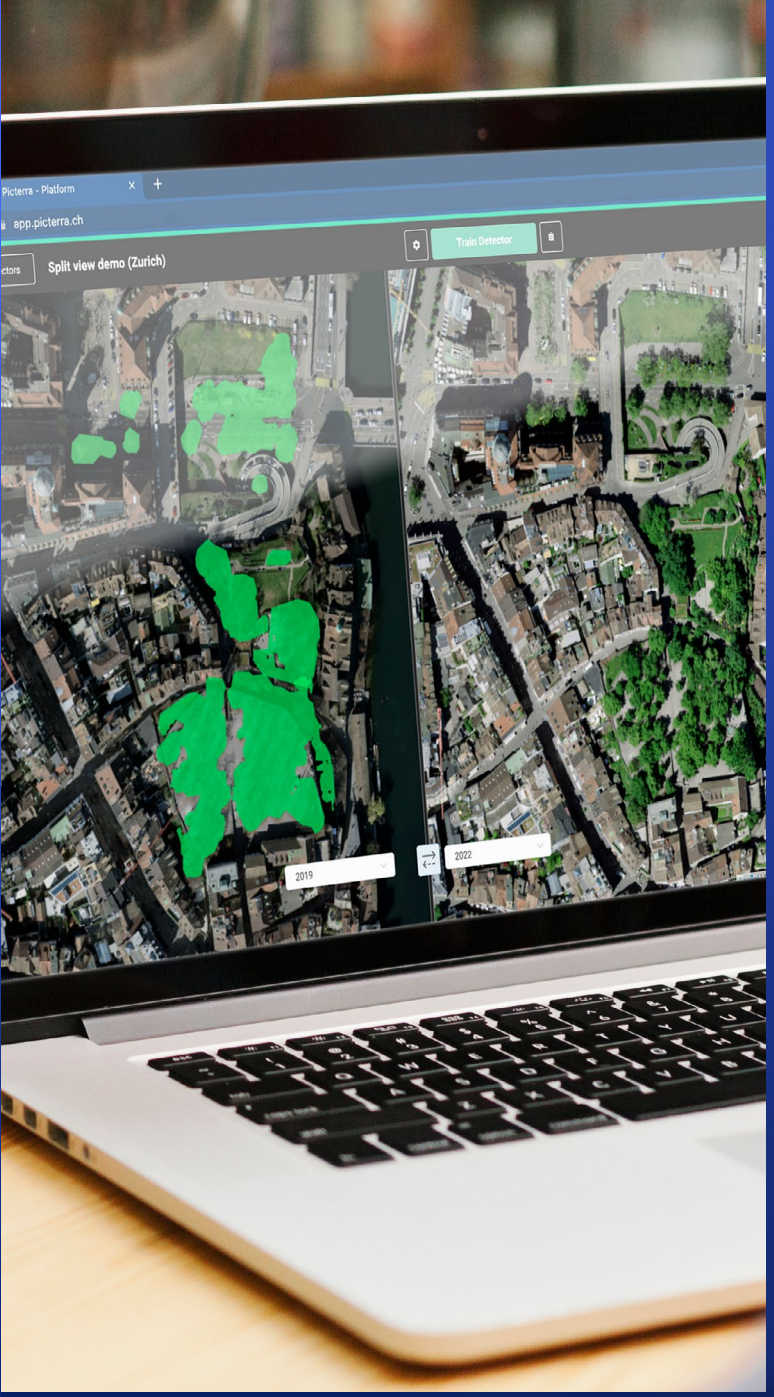
Picterra

What is change detection?

Navigating change detection with Picterra

What is change detection?

Change detection might seem like a straightforward concept at first glance, often synonymous with “something that has changed.” Semantically, it appears simple, but from a technical perspective, it unfolds into a realm of possibilities, each defining a potential solution in its own right.



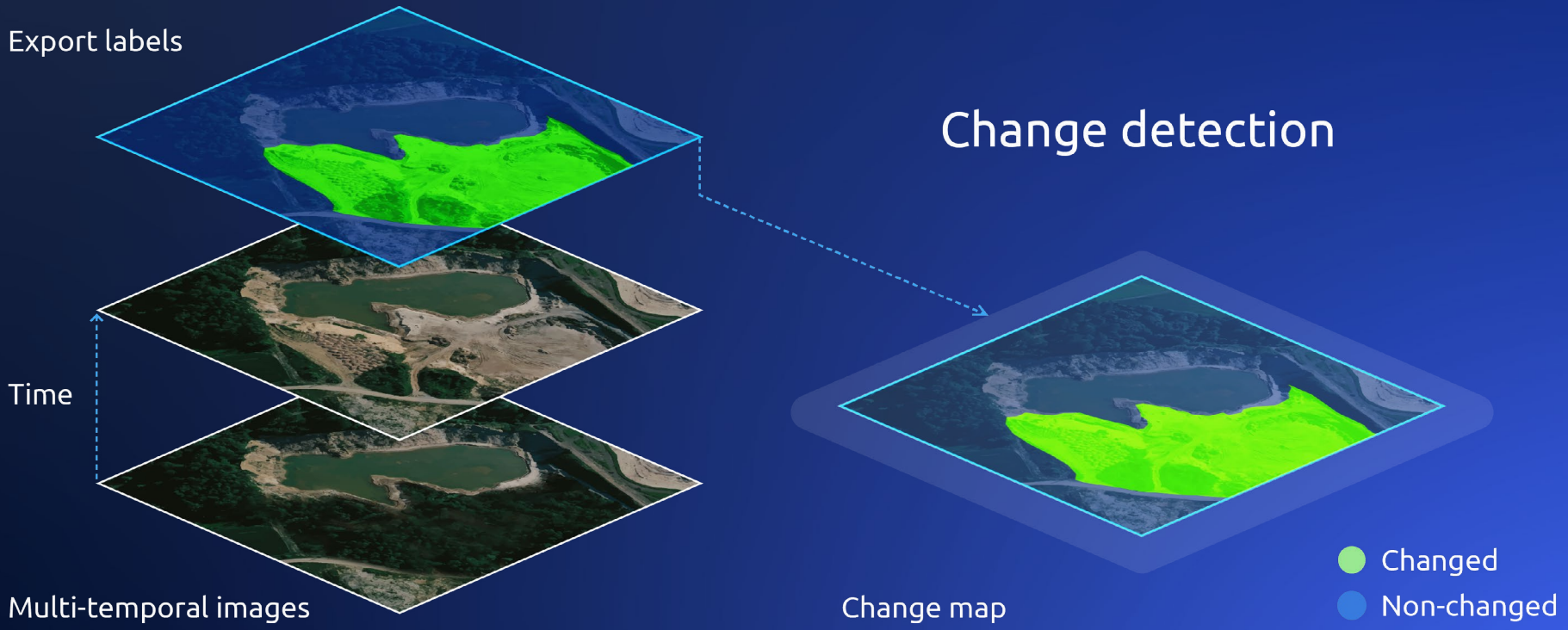
The nature of change: What do you seek?

It's easy to say you want to find anything that changed, but be careful what you wish for. The context of change detection hinges on the imagery and its intended application.

Let's delve into a scenario with two images capturing the same location on two different days. It's quite possible that you will consider the two images to look the same but at the pixel level, everything could be completely different. Maybe you took the second photo at a different time of day and everything is slightly darker. Consider a snapshot of a forest on a windy day—within moments, the landscape transforms, potentially rendering the images distinct.

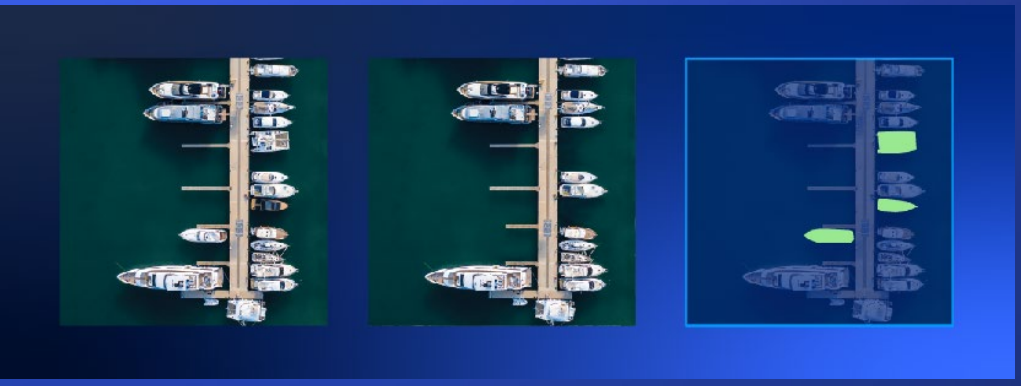
Now, envision a scenario where your focus is solely on changes related to specific elements—roads or buildings, for instance—disregarding the dynamic fluctuations of the trees. Even if the trees themselves undergo alterations, your attention remains directed towards these designated features. Here lies the crux of the matter: precision in defining the scope of change detection becomes paramount. Without detailed specifications of the problem at hand, a viable solution remains elusive.

Change detection in remote sensing



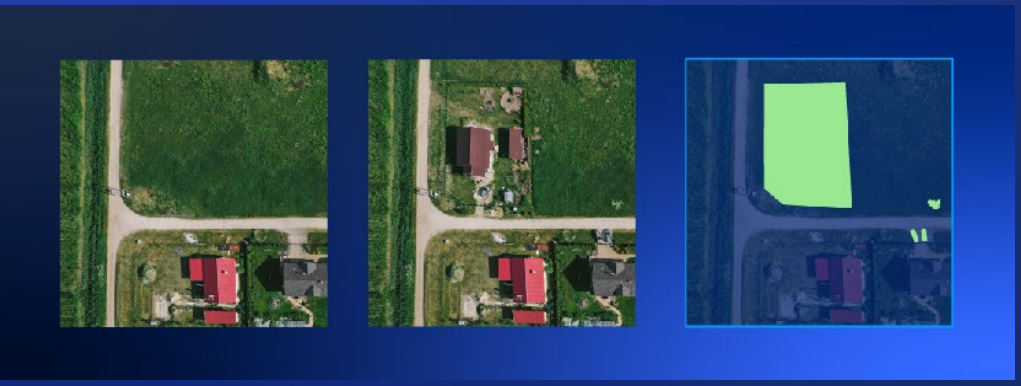
How can we define actual change scenarios?

Let's look at some real situations where change happens:



Something is disappearing

You are focusing on a well-defined object or texture in the imagery and you want to detect when it is disappearing. For instance, in the case of specific objects, this could be, for example, individual trees being cut down. At the texture level, this could be patches of forest, depending on the resolution of your imagery. Some other examples could be buildings being demolished (or part of a building), or water bodies shrinking.



Something is appearing

A bit self-explanatory but the opposite of something disappearing. Trees are being planted and growing, buildings are being constructed, and perhaps new oil spills or construction material piles appear between two dates.

Something is shifting/moving

An object is moving - this scenario is especially relevant for infrastructural elements. For example, on railroads, there are certain connective pieces that are not supposed to move, but just due to natural wear and tear, they can shift locations. It's important to detect these shifts and alert maintenance crews to avoid any unfortunate accidents caused by these changes.

Something is changing appearance

Finally, it is not about things neither appearing or disappearing but just changing in appearance. For instance, think of a building under construction - between two dates the top-down appearance of the building changes for example due to parts of a roof being added. Another example related to plant life - you could explore automatically detecting changes in the NDVI signal of a crop, indicating shifts in its vitality and vigor.

How can Picterra tackle change detection use cases?

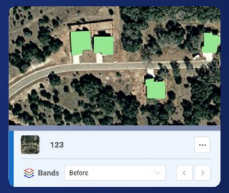
Picterra offers two potential ways to cater to your specific change detection needs; the *comparative change detection analysis* and the *change detection model*.

Change detection solutions

01 Comparative change analysis

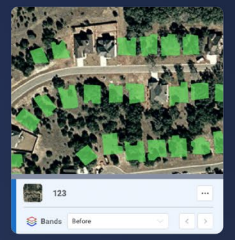
Before image

After image



Objects or textures detections

Results



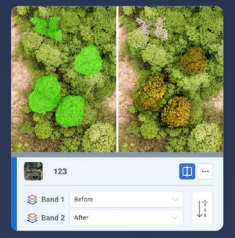
Change detection through comparison of generated detections

02 Change detection model

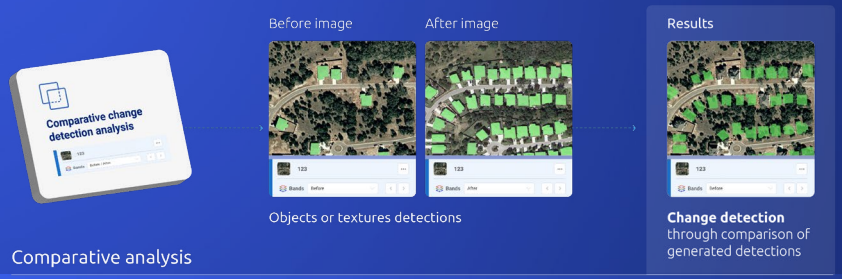
Combined before and after images

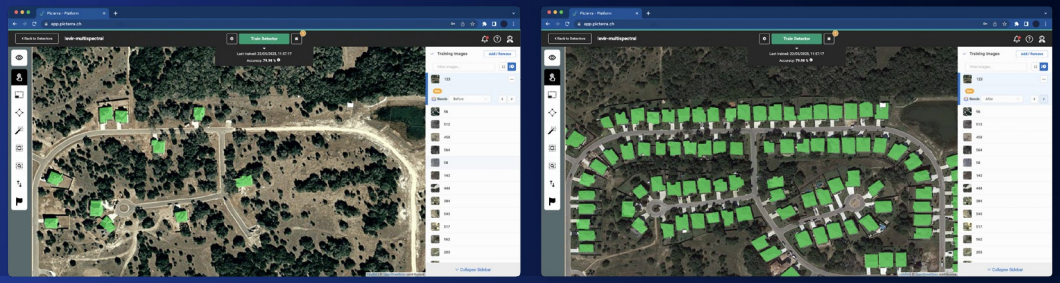
Changes annotations across paired images

Results



Change detection through single model output

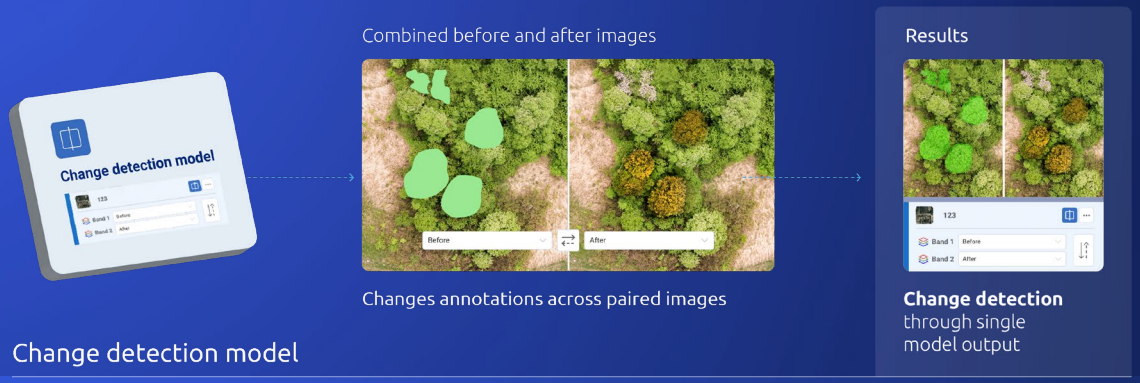




The comparative change detection analysis

In a given scenario with two images, we initiate the process by detecting an object/ texture in both the “before” and “after” images. In the next step, we directly compare the two sets of detections. For disappearing objects, if the detection exists in the first image but not the second then that counts as a disappearance. For appearing objects, if the detection exists in the second image but not the first then that counts as an appearance. For moving objects the process is similar but with additional consideration. You have to make a decision about whether or not something has moved based on how much overlap there is between the locations of the two objects. There is a bit of a blurry line here since if the object has moved so much that there is no overlap at all it may look like one object has disappeared and another has appeared. To address this, you can either do a “search” within the radius of the detection of the first image to match the detection in the second image or more generally incorporate all three possibilities into a single “change” indicator which can be turned into a localized alert for manual review. In Picterra this detection comparison is implemented as an advanced tool with customizable parameters to tune to suit your use case. It’s worth noting that this geometry comparison doesn’t involve machine learning; it’s simply about comparing geolocated polygons generated by Picterra detectors.

The effectiveness of this approach largely depends on the accuracy of individual image detections. If you’re dealing with numerous false positives or false negatives, you may experience a rise in misleading alerts about objects disappearing or appearing. However, this method truly shines in scenarios where you’re confident in the accuracy of detections, such as with high-resolution railway imagery where standardized backgrounds provide a strong foundation for reliable detections. Nevertheless, it’s important to note that this approach doesn’t account for the “change of appearance” scenario. To address this aspect, let’s turn our attention to the change detection model.



Change detection model

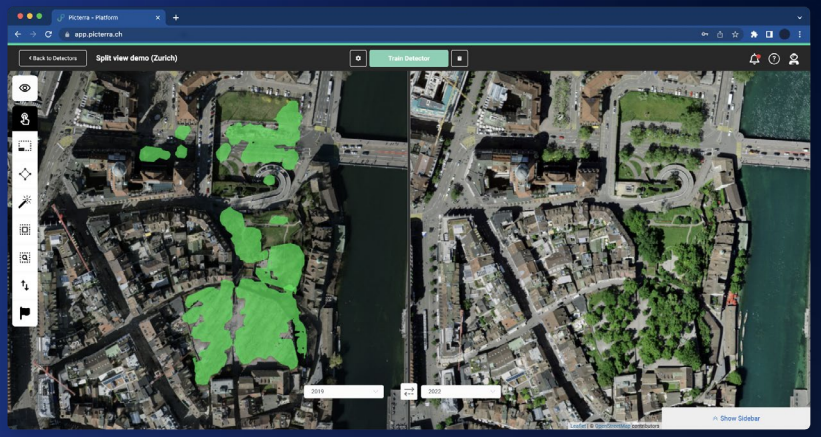
The change detection model

The second approach shares a conceptual similarity with our custom detector. However, instead of annotating on a single image individually, you annotate with the view of both images together—the “before” and “after” images—with your cursor visible simultaneously in both. This approach allows you to easily annotate the change itself across the two images if an object transforms between them. Alternatively, you might choose to annotate only a particular category of change. For example, you could focus on identifying a building’s transition from its foundation to being fully constructed (including a roof) but exclude changes like adding solar panels to an existing roof. This selective annotation enables you to concentrate solely on instances that match your predefined pattern or definition of change.

After annotation, you can initiate training by clicking the “Train” button as usual. The key difference between training a custom detector and a custom “change” detector lies in the input. In this case, you’re training the machine learning model using both images combined, along with your annotations of the change between them—not just annotations on a single image. Subsequently, when using the model for detection, the input remains the combined pair of images.

Picterra offers dedicated tools to streamline this workflow. The first one - “Format images for change detection” - is an advanced feature that merges two images—the “before” and “after” — into a single input image for both training and detection. The second tool, the “Split view,” serves as an annotation tool. The split view divides your imagery into two halves: one side displays the “before” image, while the other side displays the “after” image (you can define their order). Annotations made on one side automatically appear on both images. This eliminates the need to switch constantly between the two images, making it easier to annotate the “change” between them.

This method allows you to annotate not only changes that involve objects disappearing, appearing, or moving but also offers the flexibility to decide exactly what to annotate based on your needs. An additional advantage of the Change detection model approach is its efficiency. Since you’re running a single model on a pair of images, it can be faster compared to running a model twice—once for each individual image. On the surface, this approach might seem superior due to its one-size-fits-all nature. However, it does have a limitation. When you’re annotating only instances of change, and if such instances are sparse in your imagery, your dataset might be insufficient for the model’s ability to learn effectively. This is in contrast to the Comparative change detection analysis where you annotate the object itself, which typically provides more examples (and thus higher confidence) in individual image detections.



In practice, though the amount of data you actually need really depends on the use case, so it’s a good idea to try either solution to see which one fits your use case the best in terms of accuracy. Our customer success team is also readily available to support you in making the right choice.

Exploring other routes to detecting changes

With that said Picterra's R&D team is actively exploring other avenues for change detection beyond what we have covered so far. Let's take a brief look at a few of these possibilities:

Coarse grid-based change detection

Sometimes, the change you're seeking isn't easily defined—consider shifts in forest density, for instance. Annotating such changes strictly with polygon tools can prove challenging. A different workflow could involve segmenting the imagery into a grid and annotating each grid cell with a "yes" or "no" for change detection. This approach yields more data but sacrifices the precision of annotations and detections. If your objective is to create a rough heatmap highlighting areas of desired change, this alternative could be a valuable option.

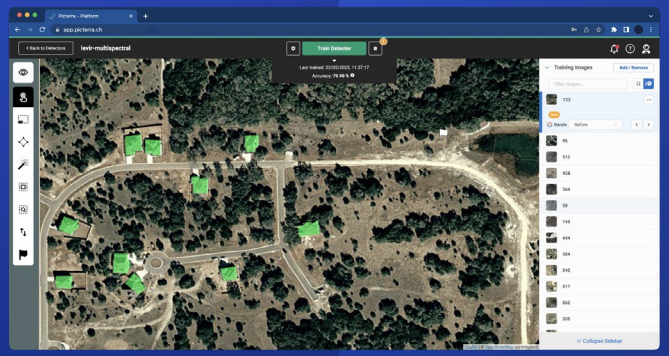
Anomaly detection

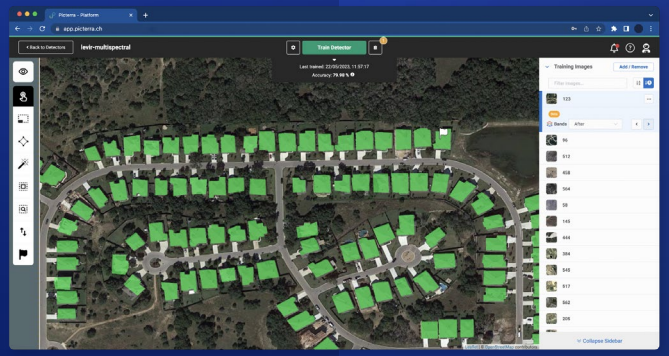
Another form of change detection is to look for any object that is unusual in your scene. In this case, there might not be sufficient data to train a model to understand what you are looking for as it may be something that has never been seen before by the model. The challenge here lies in training the model to understand what "normal" looks like and then detect anything that doesn't fit that definition of "normal". The difficulty of this approach is that the more varied your definition of "normal" is, the more data you will need. For example, identifying litter on gray asphalt appears feasible - the "normal" is just "gray and flat" so anything deviating from that can be considered an anomaly. However, in a densely populated urban environment, anything could be an anomaly, rendering the task notably more complex. This anomaly detection approach can also be applied in a grid-based manner, where you classify each grid cell as being normal or not.

Change over time series data over a fixed AOI

While we've mainly focused on techniques involving pairs of images, time series data introduces an additional dimension. Multiple imagery dates offer context about what does or doesn't qualify as change. This richer temporal data provides insights into the pre-change and post-change states, making the model more resilient to factors like lighting and weather fluctuations. It also enables strategies such as using only cloud-free dates, a significant advantage in regions where cloud cover can be restrictive.

Furthermore, within a fixed area of interest (AOI), one could envision developing a model tailored specifically for that AOI. This means customizing a model to detect change within a small, defined subregion. Rather than designing a model to scale spatially, you're crafting a model for a specific zone, simplifying the learning process. While this might entail less spatial data for training, leveraging an abundance of time series data compensates for this limitation, enabling your model to adapt effectively.





Concluding thoughts

Change detection is undoubtedly a multifaceted subject, far from being as straightforward as a simple directive to “find anything that changes.” Its intricacy is evident, given the multitude of diverse approaches available, the ideal choice of which really depends on your use case, your desired output, and your specific workflow.

Picterra currently provides two potential solutions—comparative change detection analysis and change detection model approach. Yet, it’s important to recognize that there exist several additional approaches worth exploring, the priority of which depends on our customer’s needs and requests. A common thread between all these approaches however is customizability. Similar to the custom detector, off-the-shelf pre-trained change detection solutions won’t suffice. Instead, a degree of user input is necessary to fine-tune and customize the solution according to the specifics of a given use case.

Our research and development roadmap evolves in response to the feedback we receive from our valued clients. If you believe certain change detection scenarios hold greater relevance than others, we encourage you to get in touch. We’re more than willing to engage in detailed discussions and explore how to best align our solutions with your needs.

Find out more at

picterra.ch