STTORM-CD: Supplementary Information

Jonáš Herec[1,2,*], **Jan Sedmidubsky**[2], **and Rado Pitoňák**[1]

[1]Zaitra s.r.o., Brno, Czech Republic
[2]Masaryk University, Brno, Czech Republic
[*]jonas.herec@zaitra.io

ABSTRACT

This supplementary information accompanies the STTORM-CD article. It provides a critical, in-depth assessment of existing tile-wise change detection evaluation methods, leading to the introduction of **two new metrics – Area under recall curve (AURC) and Required Downlink Percentage (RDP)**. This document also includes a survey on **onboard georeferencing**, an unsolved problem that is essential for effective onboard change detection. Additionally, supplementary tables and visualizations are presented to provide a thoroughly exhaustive assessment of the results.

Contents

List of Figures

List of Tables

1 Metrics

This chapter will introduce the RaVAEn [1] evaluation method and show why it brings biased results. Following up, new metrics will be proposed.

1.1 RaVAEn evaluation and its problems

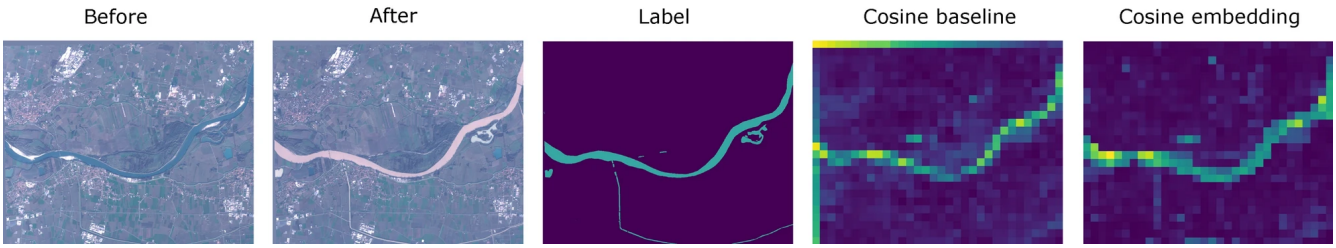

Figure 1. RaVAEn evaluation system. Heatmaps on the right, capturing the change predictions, are compared against the ground truth mask in the middle. Figure taken from RaVAEn article [1].

To evaluate their models, RaVAEn created heatmaps based on predicted cosine distances, as shown in Figure 1, and compared them with the ground truth masks. This means assigning a cosine distance value to each pixel in the heatmap. If a tile had a cosine distance of 0.4 compared to the reference tile, all pixels in that tile would be given a value of 0.4. This keeps the resolution of the heatmaps the same as the ground truth labels. These heatmaps were then checked against the ground truth masks using the Area under the Precision-Recall Curve (AUPRC). The Recall and Precision were defined as shown in Equations 1 and 2. For each disaster dataset (RaVAEn-Floods, RaVAEn-Wildfires, etc.), the reported statistic was the mean AUPRC, where the AUPRC was calculated for each time series individually, and then the mean was computed.

$$\text{Recall} = \frac{\text{Retrieved changed pixels}}{\text{All changed pixels}} \tag{1}$$

$$\text{Precision} = \frac{\text{Retrieved changed pixels}}{\text{Retrieved pixels}} \tag{2}$$

Using the area under the precision-recall curve is a good idea because it avoids setting a specific threshold for tile classification. This is particularly important in this task, where the focus is more on the order of tiles rather than strict classification, so assessing how well our model performs at various thresholds provides more information than selecting the best-performing threshold and computing the F1-score or similar metric. However, evaluating the AUPRC on a per-picture basis and then averaging is not ideal. This approach involves adjusting the threshold to change recall levels, which may not accurately reflect real-world conditions when applied individually to each picture. Instead, it should simultaneously be applied to all images in each dataset. This simulates real use more, where we would decide the best classification threshold beforehand, and it would stay the same for all the images taken.

But, in this specific case, relying on precision and recall impairs the score. Consider the results obtained for different disaster types: RaVAEn achieved the highest AUPRC of 0.913 for wildfires, while the lowest AUPRC of 0.448 was obtained for floods. When describing the dataset, it's noted that while a similar number of locations represent each type of event, the affected area varies significantly depending on the disaster type. Specifically, the RaVAEn-Wildfires dataset exhibits the largest area of effect and the largest proportion of changed pixels. This observation appears to be more than coincidental.

Wildfires tend to spread uniformly, consuming everything in their path. As a result, they often leave behind extensive burned areas, meaning that many tiles in the wildfire dataset have nearly all pixels marked as changed in the ground truth. In contrast, floods spread differently. Water flows into lower-altitude areas and may accumulate in certain spots while receding elsewhere. Consequently, the impacted area by floods is often more dispersed than that of wildfires. In many cases, the signs of a flood may not occupy the entire 32x32px area of a tile. This leads to floods producing numerous tiles containing flooded pixels to some degree, but rarely are all pixels in a tile marked as changed. If we apply RaVAEn's method to floods, it could significantly underestimate the model's performance. Even more important is that when comparing two models across the same disaster, the difference in performance is diminished by a priori given precision.

This impairment will be demonstrated through a fictitious example featuring two images: one of a wildfire and one of a flood. Each image contains 10 tiles with changes, with the wildfire tiles having 90 % of their pixels changed and the flood tiles having 30 % of their pixels changed. Two methods were used: a baseline model and a trained model. For simplicity, assume

that neither model makes any false positives, but they differ in their false negative rates. At the same threshold, the trained model correctly identifies 8 changed tiles, while the baseline identifies 2 changed tiles.

At this threshold, the recall for the baseline is 20 %, and for the trained model, it is 80 %, showing a significant improvement of the trained model over the baseline in both cases. However, when calculating the AUPRC in RaVAEn, this improvement is not accurately reflected. This is because precision is determined by the number of changed pixels in the tiles, which is 90 % for the wildfire tiles, and 30 % for the flood tiles (after the threshold). An illustration of this case can be seen in Figure 2. In the wildfire case, the AUPRC for the baseline is 0.91, while the trained model has an AUPRC of 0.94. For floods, the baseline's AUPRC is 0.37, and the trained model's AUPRC is 0.58. In both cases, the 60 % improvement is diminished by the precision, which can not be changed by the model, but it is given.

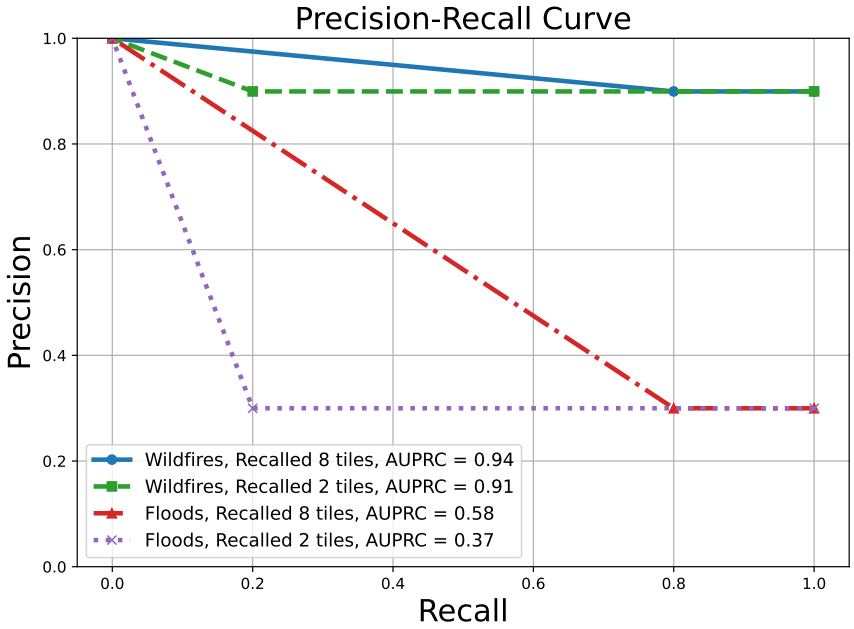

Figure 2. Illustration of RaVAEn evaluation method and its impairments. The same difference in model performance yields very different results across the disasters because of a given change distribution in the dataset.

As a result, the dataset structure heavily influences the metrics, failing to evaluate the model performance. The only thing that can be read from them is the binary assessment of whether the model performs better than the baseline on a given disaster.

1.2 Proposed primary metric – AURC

Here, an alternative approach will be proposed to address this impairment. As previously mentioned, the change detection system operates fundamentally as an ordering system. When a disaster occurs, we prioritize identifying the most severely affected areas, followed by the peripheries. One might argue that this resembles a regression problem, where we could measure the correlation between predicted change and the proportion of changed pixels. While this approach isn't entirely incorrect, it overlooks the essence of the task. Correlation indicates whether there's some degree of ordering present, but it doesn't provide insight into recall or precision. In simpler terms, it doesn't inform us about how many significant areas we've successfully identified versus those we've missed.

We instead propose another solution: the tiles will be ordered by predicted change, and only the upper portion of the predictions will be used to calculate recall on a per-pixel basis. For example, it could answer the question: If the top 10 % of tiles, when ordered by the predictions, will be retrieved, how many changed pixels will it retrieve?

For instance, if the top 10 % of the tiles ordered by ground truth contain 100 changed pixels, but the top 10 % of tiles, when ordered by the predictions, only contain 78 changed pixels, the recall score would be 78 %. This method focuses on the capability of the predictions to order the tiles, but does it more practically than correlation. Because it directly tells us how many pixels we have actually retrieved from those that could have been retrieved. However, this approach still presents biases. For instance, in the RaVAEn article [1], the flood dataset has 6.74 % of pixels classified as changed, while the wildfire dataset has 53.79 % of pixels classified as changed. Using a fixed threshold, e.g., top 10 % of tiles, might not make sense — this could unnecessarily include unchanged tiles for floods and too few changed tiles for wildfires, failing to provide informative insights about the remaining changed tiles.

That's why we have modified this method further. The recalled percentage will be calculated solely from changed tiles. For example, if a dataset contains 500 tiles in total but only 120 of them contain changed pixels, retrieving 100 % means retrieving the 120 most changed tiles when sorted by our change prediction, and retrieving 10 % means retrieving 12 tiles. This adjustment serves two purposes. Firstly, it ensures fairness by allowing a perfect system to retrieve all changed tiles and nothing more. Secondly, this approach maintains fairness regardless of the change distribution across tiles. If there are a lot of changed tiles with a big proportion of changed pixels, a fully changed tile will have a small influence on the result. Conversely, if there are only a few changed tiles and they contain only a small number of changed pixels, each tile will have a significant influence on the metric. This allows for accurate comparison of the models within various datasets with different change distributions.

Moreover, in real-world usage, we might download a different portion of tiles than just 10 %. Therefore, arbitrarily selecting this percentage or any other value doesn't accurately reflect the practical use case. It could also become a testing "hyperparameter", which, instead of truly reflecting the model's performance, is being tweaked to achieve visually appealing results. That is why the final modification is calculating recall for all one hundred percent and then using the area under the curve to derive the final metric. This gives us an accurate depiction of model performance for downloading various amounts of data. This system is illustrated with a fictional example and larger steps between the retrieved thresholds in Figure 3.

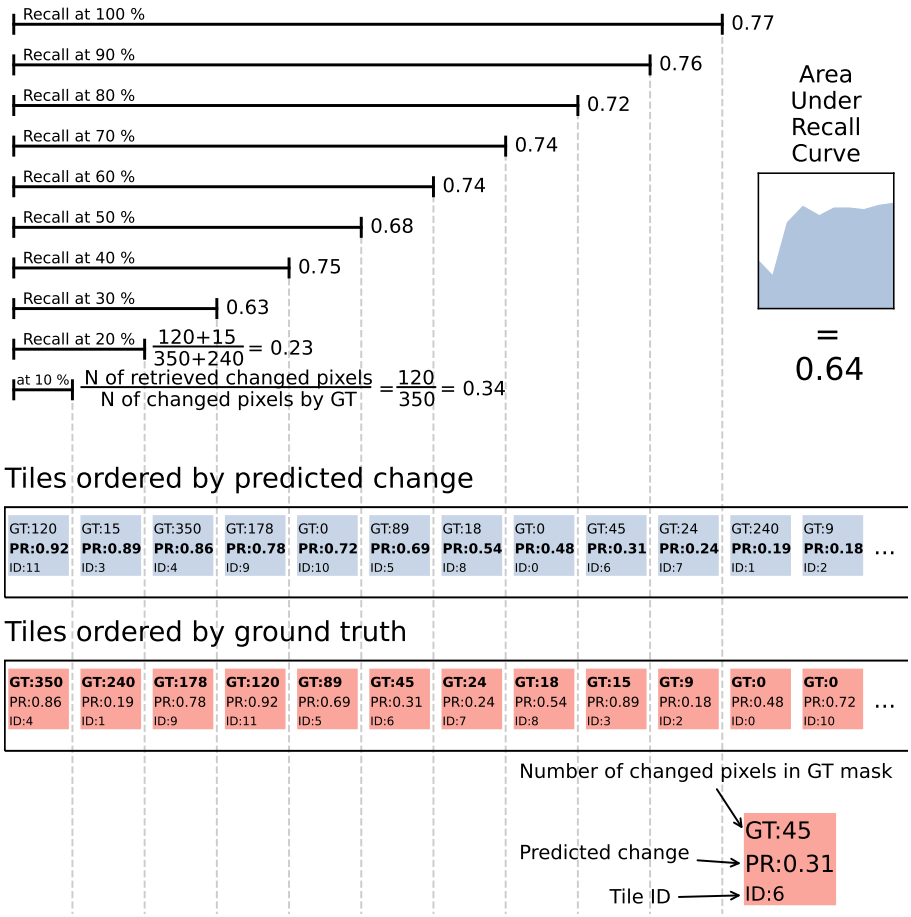

Figure 3. Proposed primary metric - AURC: The actual step size, which we use is 1 %, but for simplicity, it is illustrated with a step size of 10 % for a fictitious, very small dataset that contains only 10 changed tiles.

This approach ensures a fair comparison of methods within a single dataset while providing interpretability and accuracy that mirrors real-world scenarios. Some might argue that excluding precision from the evaluation is flawed, as it overlooks a comprehensive assessment of method efficacy. However, implicit precision is inherently embedded within this approach because including unchanged tiles diminishes the recall, and this effect is inversely proportional to the number of retrieved tiles. In top percentages, where almost all tiles are fully changed, including unchanged tiles can be costly. Inversely, when retrieving 100 % of changed tiles, the tiles at the bottom have only a few pixels changed, and including unchanged tiles is thus less costly. This is in sync with our use case where the more changed tiles are more important.

Moreover, literal per-pixel precision holds less significance in this context. The primary objective is to identify and retrieve tiles with changes, prioritizing those with more significant changes, such as fully flooded areas, over partially flooded or unchanged areas. Hence, the emphasis lies more on the portion of change captured rather than the number of unchanged pixels spared. Additionally, considering that catastrophic events are infrequent, it's highly likely that satellite data used for disaster detection also serves other purposes. Therefore, downloading a tile with no change doesn't necessarily have negative implications as long as we've successfully captured most of the change.

However, despite its alignment with real-world applications, interpretability, and fair comparisons within a dataset, it's crucial to acknowledge that this approach doesn't guarantee equitable comparisons between different disaster types. The conditional probability of tiles within the changed region being changed remains higher for datasets with greater change proportion (wildfires). However, this effect should diminish with larger and more balanced datasets, as well as with a more accurate change detection model, where randomness should play a smaller role. However, treating each disaster type as a separate dataset and comparing the distance between baseline and model performance, rather than solely comparing absolute metrics, is essential.

1.3 Proposed secondary metric – RDP

The proposed approach, however, can miss one important information. How much data do we need to downlink to receive all changed tiles? This is particularly an issue when the size of the disaster differs. For example, in the case of a small island, the disaster could hit the entire island, which is quite important information. However, it does not have to be reflected in the proposed metric because only a few tiles have been changed. Such is the case of the RaVAEn-Hurricanes dataset [1]. It has one big event with many changed tiles and then smaller events. If the model performs well on the big event, the performance on the smaller events does not affect the metric significantly.

That is why a secondary metric will be used. It is very simple – it measures the amount of data needed to downlink to receive all changed tiles. For this, we must define what changed means, which is problematic. We are probably not interested in tiles containing only a few pixels of change. However, we don't want to miss even the small changes. So, we will use 5 % as the threshold. In the case of 32x32 px tiles, this is roughly 7x7 px, representing app. 70x70 m area. This allows us to focus on tiles with a small but meaningful change. When used with the primary metric, which focuses more on the top changed tiles and the sheer amount of change, it can complete the picture of the model's performance.

2 Onboard georeferencing – quick survey

For the onboard change detection system to function, the current image needs to be matched quite accurately with an image from the past. To achieve this, some level of georeferencing is required. This remains an unsolved challenge, and surprisingly, none of the onboard change detection studies (cited in the main text) address this issue. The geolocation derived from the satellite's GPS is often insufficient for precise georeferencing. On Earth, this problem is typically resolved using indirect georeferencing, which involves aligning a georeferenced reference image with the raw image. However, storing such reference images onboard the satellite is not feasible due to the limited onboard storage space.

There are a few proposed solutions currently available. Most of them focus on specific parts of the georeferencing pipeline, but only [2] proposes a complete system. Their system suggests a robust, lightweight database of georeferenced features that can be matched via template matching (in this case, normalized cross-correlation) to the current raw image. First, they extract edges from the raw image using phase congruency, an edge extractor invariant to contrast and illumination, which produces standardized values between 0 and 1, making it ideal for use with various satellites and sensors. These values are then thresholded to reduce each pixel to a single bit. Run-length encoding compression is then applied to further reduce the image size. Only a subset of the image, focusing on invariant features like coastlines, roads, or airports, is stored, making the approach suitable for various satellites and sensors. This results in a lightweight database created from reference images of Earth. During flight, images are matched to reference images based on geographic position. The images are downsampled to reduce details and computational complexity before performing template matching. Harris corner points are detected, and the image is divided into patches around these points for even distribution. These patches are then matched with the reference image, with outliers removed using RANSAC. The accuracy is 100% with errors under 10m, 93% under 5m, and an average error of about 3m. Processing time on an Arm A57 CPU is 31s, 4x slower than on a 256 CUDA cores GPU (7.26s) for a 7920×1200 image, raising concerns about its feasibility for smallsats and commercial use.

Another approach is more obvious, image matching is often done through algorithms like SIFT, SURF, or BRIEF, which extract key points in the image, and each key point is described by a vector (descriptor). Then you can use Euclidean distance or kNN to match the key points from the images. This holds potential for on-board purposes as, for example, BRIEF uses a 256-bit binary vector as the descriptor. Storing 1 key point per 1 km with latitude and longitude stored as 32-bit floats for each point would require approximately 10 GB for storing the whole Earth (excluding the oceans). However, the key point extraction algorithms and matching of the key points tend to be computationally heavy, so recent research has proposed their acceleration

by doing the computation on an FPGA. Mostly, the researchers focus on fast and robust SURF - [3, 4]. But SURF descriptors consist of floats, so it takes significantly more space than BRIEF. So for our purpose, the most interesting study is the [5] which uses FPGA acceleration and SURF as a keypoint extractor but uses BRIEF descriptors and BRIEF matching, which consists of fast bit-wise XOR operations. Extraction, description, and matching of two 512x512 images took 2.62ms with the proposed method. Its score is also acceptable, with precision fixed at 100 % recall is 0.70-0.82 (70-82 % of key points were matched) for rural areas and 0.3 for bare soil. This is acceptable as only a few key points are required to match the images. However, it was tested on images from the same sensor, so the potential problem with this approach is a generalization. To have a complete solution, you need to upload a database to the satellite beforehand, created from available worldwide imagery such as Sentinel 2 data, and this approach's ability to extract the same key points on pictures from different sensors is untested.

If the performance of classical algorithms is insufficient, another possible approach would be using a neural network for key points extraction, such as D2-net [6], which can robustly extract the same features and correctly match images. The input could be a panchromatic image or an RGB image, which would make it usable across most sensors. However, for its on-board use, it would need to be compressed or replaced by a smaller network, and it is unclear if this network would perform precisely and fast enough.

3 Results

Here, the additional tables and visualizations concerning the results will be presented. Let it be noted that for all direct visualizations presented here, the scale for each method was set independently using the $min()$ and $max()$ values derived from the entire dataset to ensure accurate performance visualization.

3.1 STTORM-CD-Floods dataset

In Table 1, the hyperparameters used to obtain the best models are listed. Figures 4, 5, 6, and 7 provide visualizations of all four images in the test set of the STTORM-CD-Floods dataset. An important observation for the interpretation of model performances is that only the STTORM-CD model consistently delivers good results. The Cosine baseline and NDWI perform reasonably well in almost all figures, but they fail to detect any changes in Figure 7, where only the STTORM-CD and RaVAEn models successfully identify the floods, though they slightly overestimated by falsely identifying wet fields. This is an exception in RaVAEn's performance, as its predictions in Figures 5, 6, and 4 are significantly noisy.

Figures 8 and 9 show visualizations of AURC, obtained using $min(memory)$ and the most recent tile for change prediction, respectively. The conclusions are largely consistent with those presented in the main text, as both were derived from the same visualization, with the only difference being that the visualization in the main text was based on $avg(memory)$ predictions. The most notable performance difference is observed for the Cosine baseline, whose performance is significantly diminished when using only the most recent tile to derive the change prediction. In contrast, when using $min(memory)$, the Cosine baseline sometimes even slightly outperforms the NDWI. Another key difference is that when using only the most recent tile, the performance gap between the NDWI and STTORM-CD is much smaller.

3.2 RaVAEn datasets

Tables 2 and 3 contain results from the RaVAEn datasets derived by using $avg(memory)$ and the most recent tile for change prediction, respectively. The trends are, however, very similar to the results derived from predictions using $min(memory)$, reported in the main text. The AURC visualizations for RaVAEn datasets, derived by using $min(memory)$ for change prediction, are in the Figures 10, 11, 12, 13. A notable insight is the superior performance of the STTORM-CD model on the RaVAEn-Floods dataset across all portions of retrieved changed tiles, which is expected since it was the validation dataset. The superior performance of the NBR for the RaVAEn-Wildfires dataset is also clear. For the RaVAEn-Hurricanes dataset, which includes flood images, STTORM-CD performs the best, but only for the first 20%, after which the NDVI becomes the superior method. In the case of RaVAEn-Landslides, it is challenging to identify the superior method, as the methods' performance curves are closely intertwined, with the exception of the notably poor performance of the cosine baseline. The direct visualizations are in the Figures 14, 15, 16, 17, 18, 19, 20, and 21.

References

1. Ruzicka, V. *et al.* Ravaen: unsupervised change detection of extreme events using ml on-board satellites. *Sci. reports* **12**, 16939 (2022).

2. Wang, L., Xiang, Y., Wang, Z., You, H. & Hu, Y. On-board geometric rectification for micro-satellite based on lightweight feature database. *Remote. Sens.* **15**, DOI: 10.3390/rs15225333 (2023).

3. Yang, Z., Hu, C. & Liu, D. Fpga image stitching design based on improved surf algorithm. In *2022 International Conference on Artificial Intelligence and Computer Information Technology (AICIT)*, 1–4, DOI: 10.1109/AICIT55386.2022.9930281 (2022).

4. Du, X. *et al.* Hardware-optimized architecture of on-board registration for remote-sensing images —take surf as an example. *IEEE J. Sel. Top. Appl. Earth Obs. Remote. Sens.* **17**, 8230–8249, DOI: 10.1109/JSTARS.2024.3377663 (2024).

5. Liu, D., Zhou, G., Zhang, D., Zhou, X. & Li, C. Ground control point automatic extraction for spaceborne georeferencing based on fpga. *IEEE J. Sel. Top. Appl. Earth Obs. Remote. Sens.* **13**, 3350–3366, DOI: 10.1109/JSTARS.2020.2998838 (2020).

6. Dusmanu, M. *et al.* D2-net: A trainable cnn for joint detection and description of local features (2019). 1905.03561.

Size	Batch	LR	E	WD	Margin	Stride
small	32	3×10^{-6}	55	1.5×10^{-5}	Fixed, 0.5	32
medium	32	3×10^{-6}	55	1.5×10^{-5}	Fixed, 0.5	32
large	32	1×10^{-6}	40	5×10^{-6}	Variable	32

Table 1. Hyperparameters used for obtaining the best STTORM-CD models. Batch is the batch size, LR is the learning rate, E is the number of epochs, WD is the weight decay, Margin specifies if the margin in triplet loss was variable or fixed; if fixed, the number specifies its size, Stride represents stride used for tiling the training data.

Dataset	RaVAEn-Landslides		RaVAEn-Floods		RaVAEn-Hurricanes		RaVAEn-Wildfires	
Tiles N	626		11253		11773		27865	
Changed tiles [%]	23.64		21.70		27.57		57.89	
Metric	Primary	Secondary	Primary	Secondary	Primary	Secondary	Primary	Secondary
Geo-index	0.881	97.12	0.693	99.99	0.873	99.99	0.960	99.86
Cosine baseline	0.710	81.95	0.722	98.19	0.678	99.23	0.883	99.99
RaVAEn – small	0.867	67.89	**0.778**	**98.36**	**0.800**	99.13	0.879	99.82
RaVAEn – medium	**0.882**	70.77	0.766	98.52	0.790	**98.87**	0.887	**99.54**
RaVAEn – large	0.872	**63.74**	0.747	99.16	0.791	99.73	**0.891**	99.57
STTORM-CD – small	0.639	79.39	**0.886***	97.07*	0.628	99.99	0.837	99.98
STTORM-CD – medium	**0.826**	74.92	0.867*	96.44*	0.696	**97.71**	0.827	**98.45**
STTORM-CD – large	0.809	**74.12**	0.859*	**92.89***	**0.733**	98.89	**0.875**	99.95

Table 2. Reevaluating RaVAEn. The final change predictions were derived by using $avg()$ on the change predictions from the memory. In RaVAEn-Floods, the geo-index used was NDWI; in RaVAEn-Wildfires, the NBR; and in RaVAEn-Landslides and RaVAEn-Hurricanes, NDVI. Highlighted are the best scores for STTORM-CD and RaVAEn models for each metric. *Used as a validation dataset.

Dataset	RaVAEn-Landslides		RaVAEn-Floods		RaVAEn-Hurricanes		RaVAEn-Wildfires	
Tiles N	626		11253		11773		27865	
Changed tiles [%]	23.64		21.70		27.57		57.89	
Metric	Primary	Secondary	Primary	Secondary	Primary	Secondary	Primary	Secondary
Geo-index	0.890	96.33	0.703	99.99	0.880	99.99	0.958	99.99
Cosine baseline	0.774	64.70	0.723	98.28	0.654	98.40	0.904	99.99
RaVAEn – small	0.843	**50.32**	**0.771**	96.33	**0.775**	99.29	0.891	99.82
RaVAEn – medium	**0.856**	51.28	0.761	95.17	0.764	**97.43**	**0.901**	99.80
RaVAEn – large	0.836	52.88	0.745	**94.63**	0.769	98.90	0.899	**99.74**
STTORM-CD – small	0.629	61.02	**0.881***	94.07*	0.607	99.82	0.807	99.95
STTORM-CD – medium	0.743	60.22	0.865*	**91.22***	0.671	**94.61**	0.768	**99.48**
STTORM-CD – large	**0.799**	**54.95**	0.852*	95.50*	**0.710**	95.93	**0.853**	99.99

Table 3. Reevaluating RaVAEn. The final change prediction was derived by comparing the current tile only against the most recent tile. In RaVAEn-Floods, the geo-index used was NDWI; in RaVAEn-Wildfires, the NBR; and in RaVAEn-Landslides and RaVAEn-Hurricanes, NDVI. Highlighted are the best scores for STTORM-CD and RaVAEn models for each metric. *Used as a validation dataset.

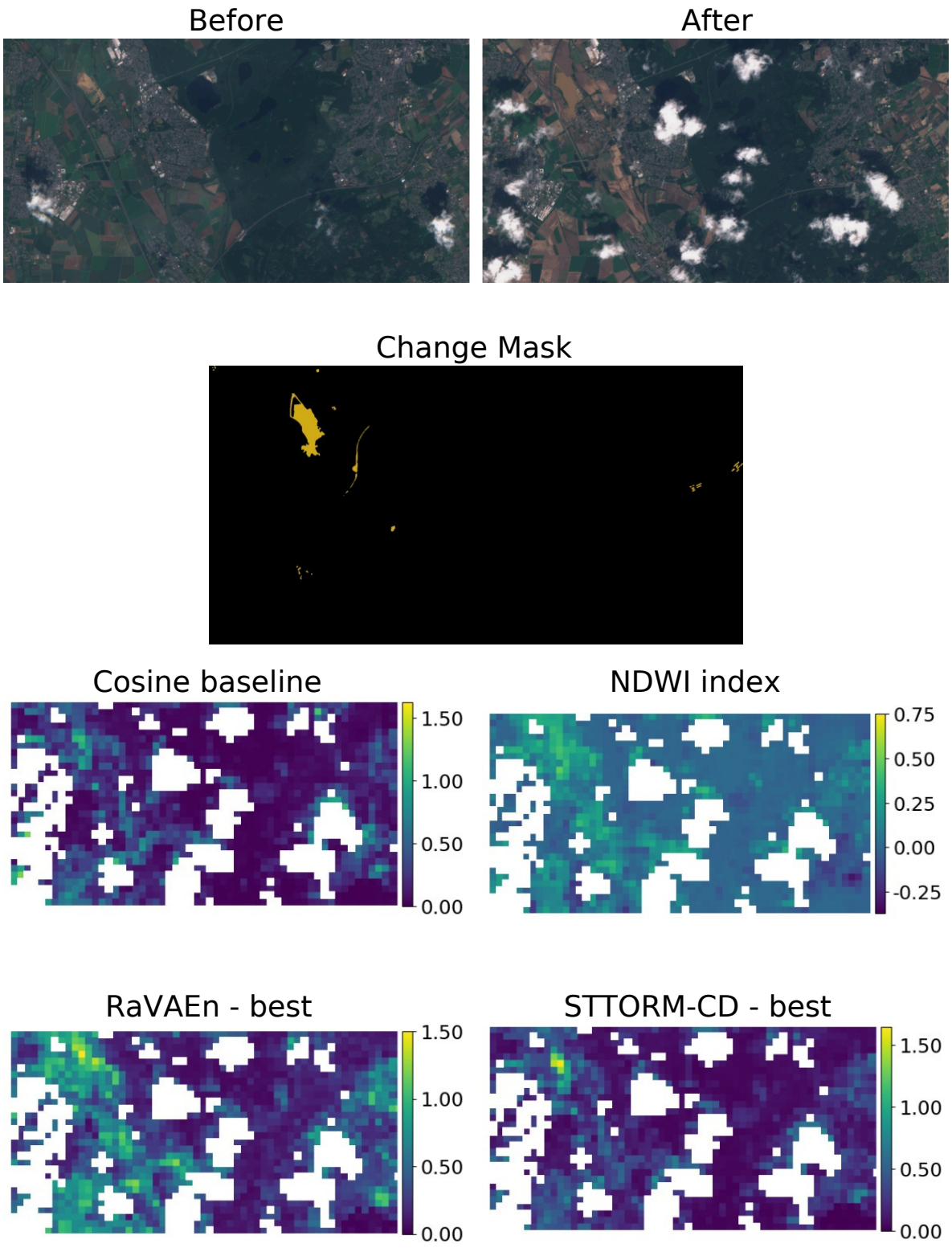

Figure 4. Germany – Visualization of method predictions. The best STTORM-CD model is the small-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $avg(memory)$.

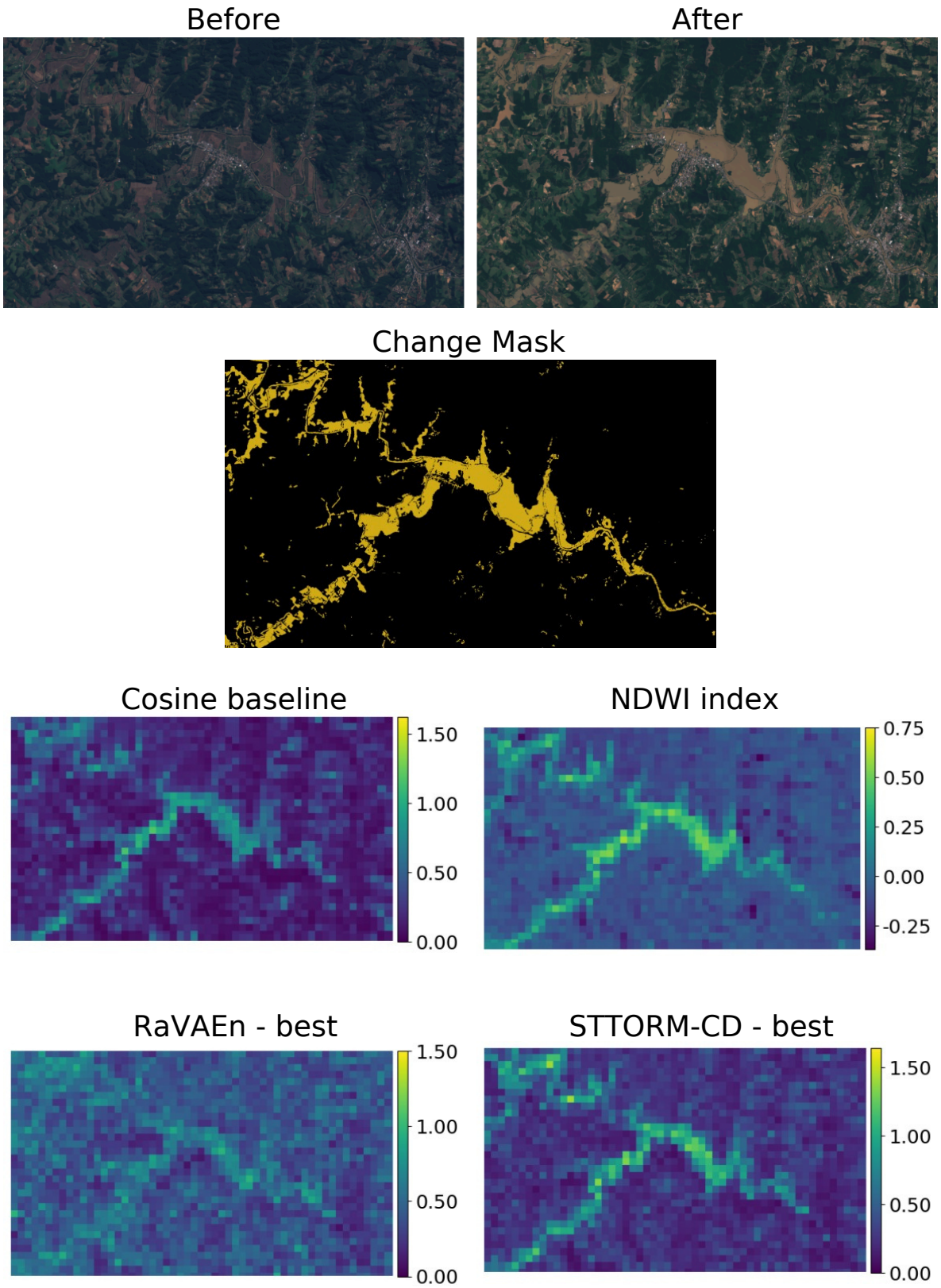

Figure 5. Brazil – Visualization of method predictions. The best STTORM-CD model is the small-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $avg(memory)$.

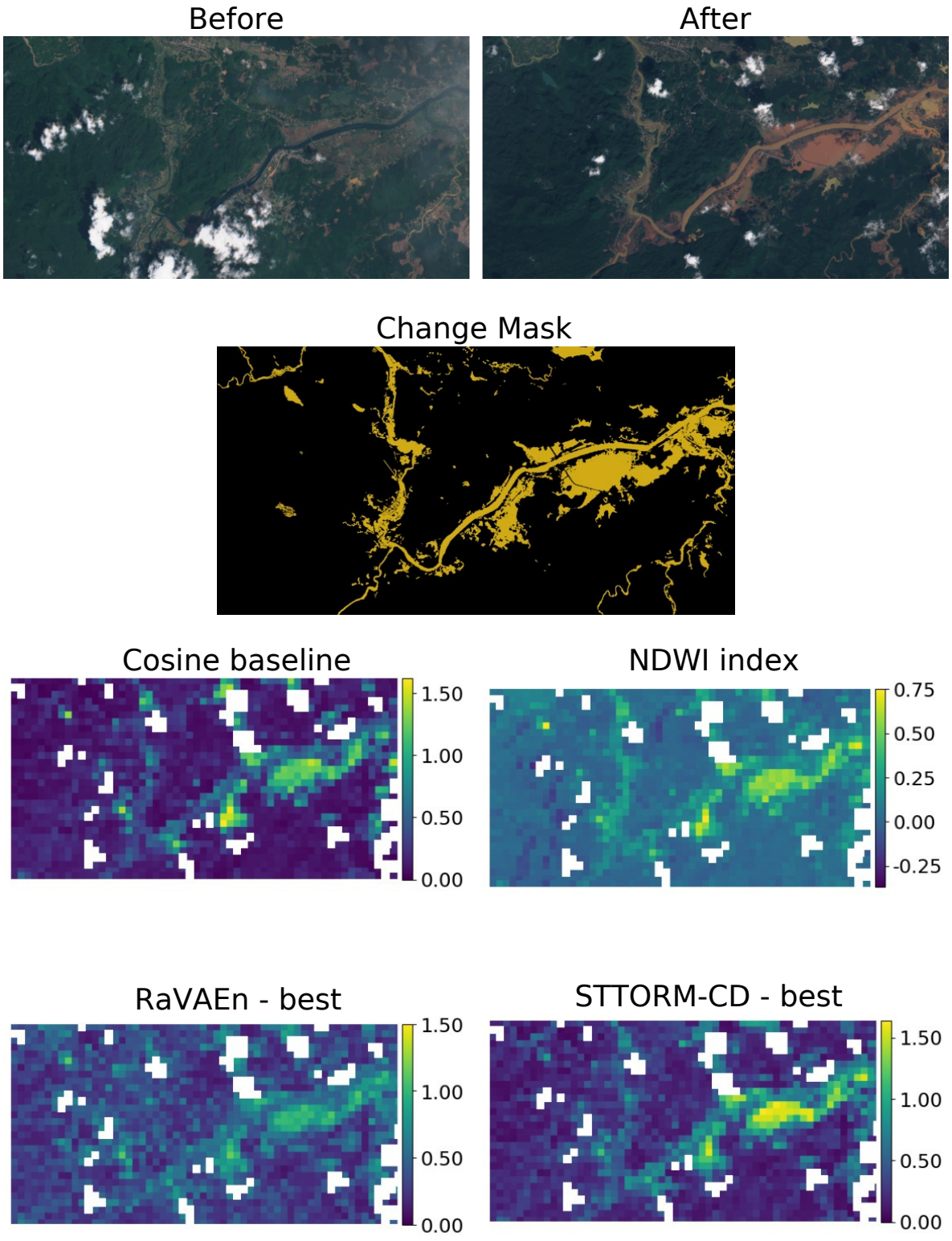

Figure 6. Laos – Visualization of method predictions. The best STTORM-CD model is the small-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $avg(memory)$.

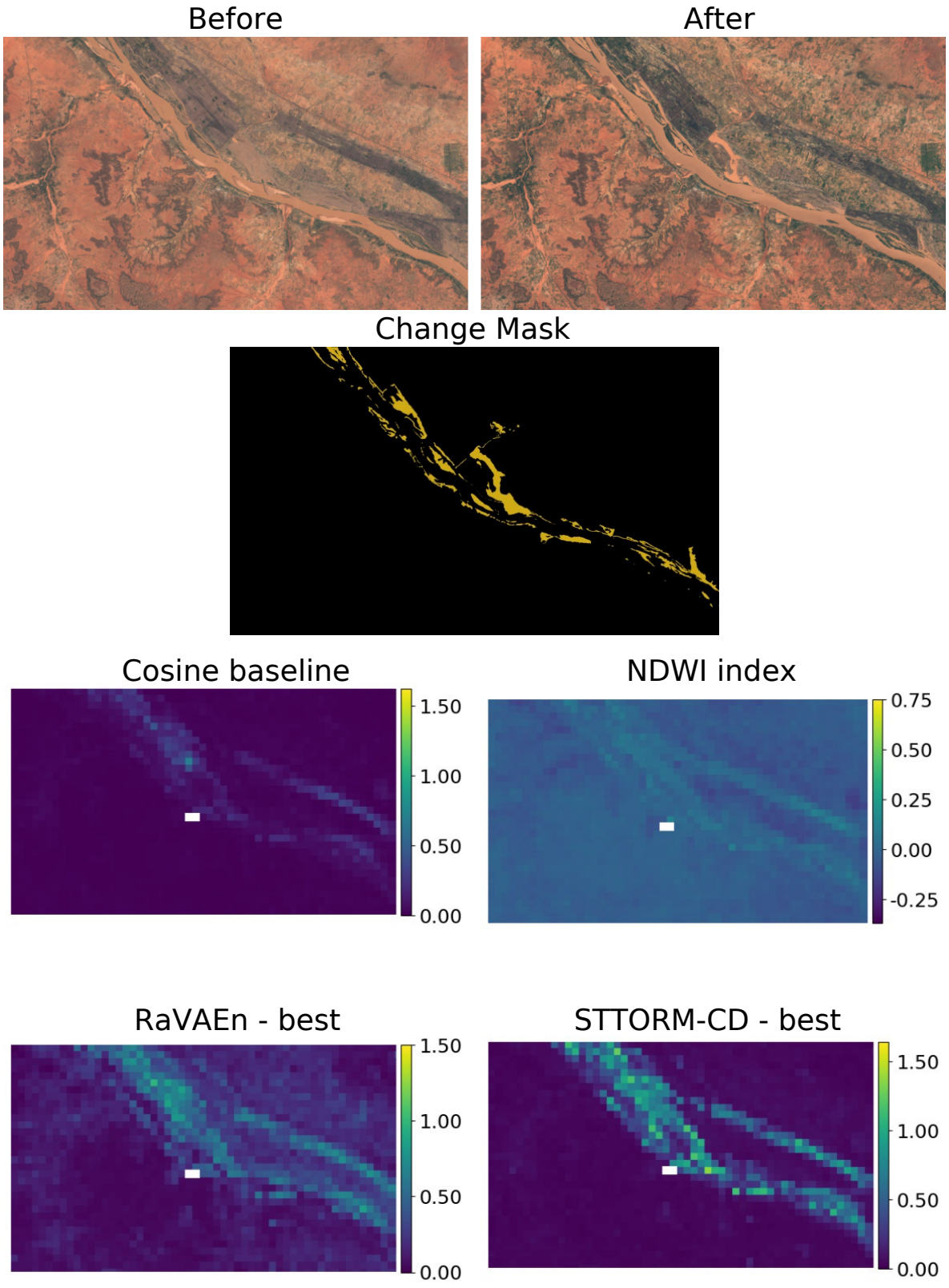

Figure 7. Niger – Visualization of method predictions. The best STTORM-CD model is the small-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $avg(memory)$.

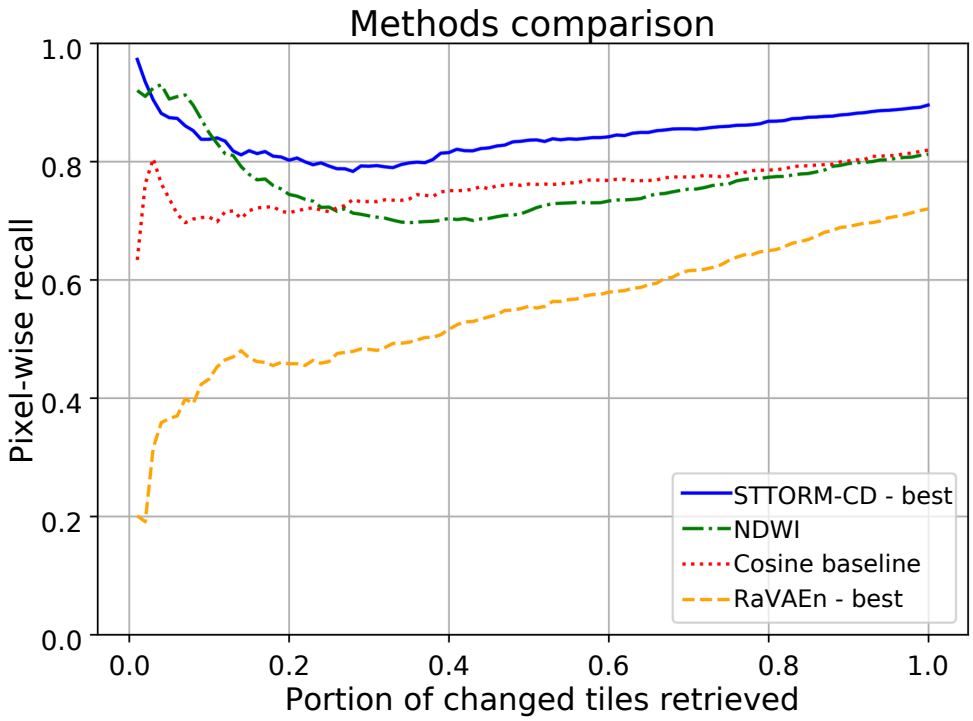

Figure 8. Methods comparison on the test set of the STTORM-CD-Floods dataset. The best STTORM-CD model is the small-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $min(memory)$.

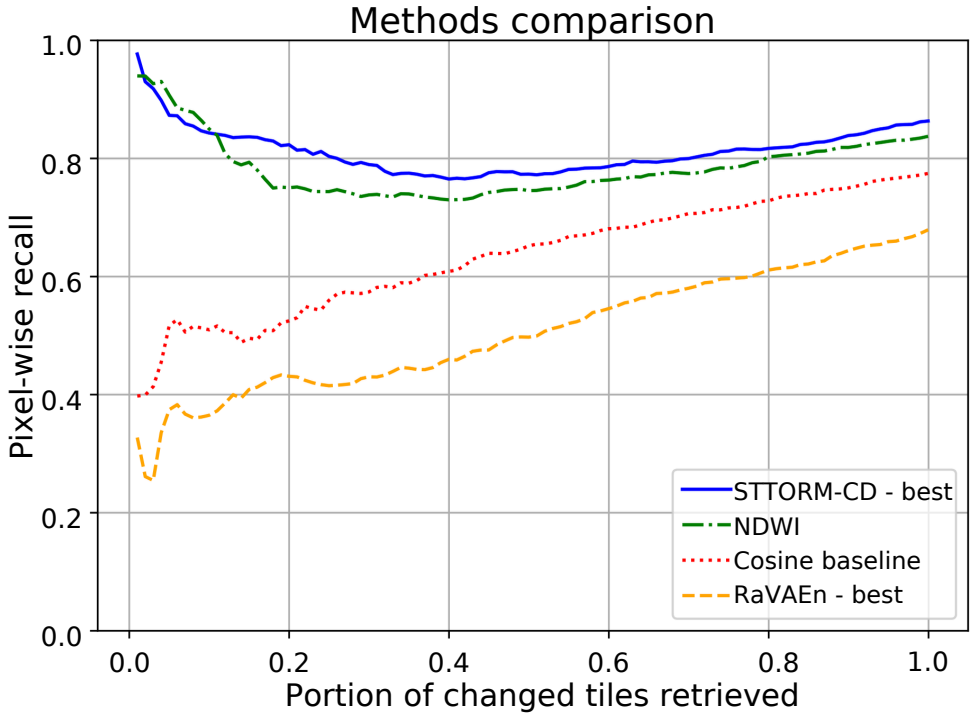

Figure 9. Methods comparison on the test set of the STTORM-CD-Floods dataset. The best STTORM-CD model is the small-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score by comparing the changed tile with only the most recent non-cloudy tile.

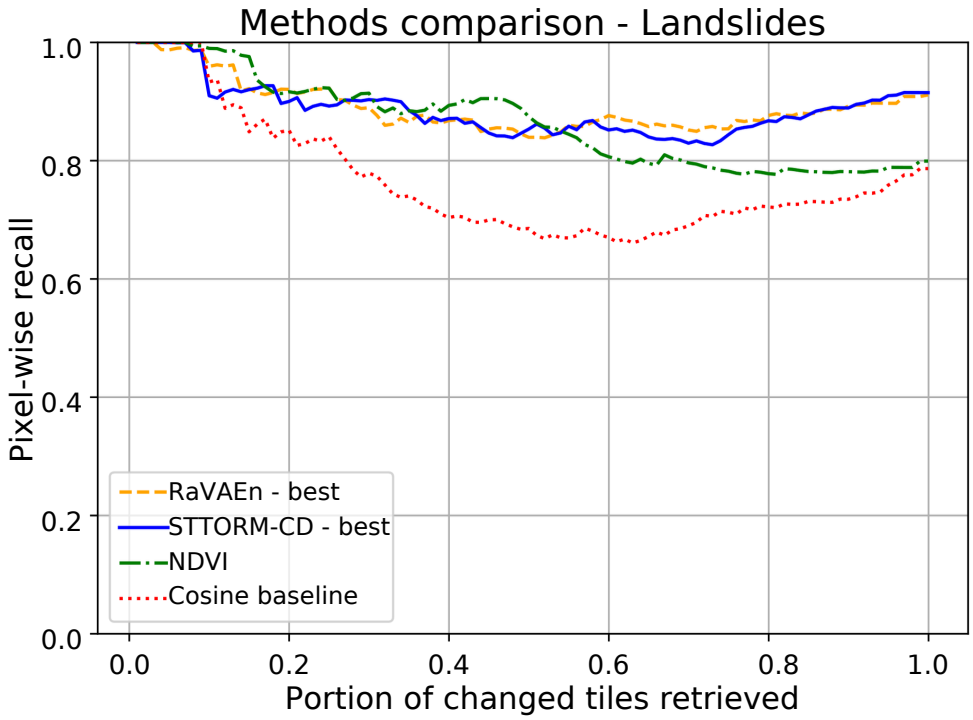

Figure 10. Methods comparison – RaVAEn-Landslides dataset. The best STTORM-CD model is the large-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $min(memory)$.

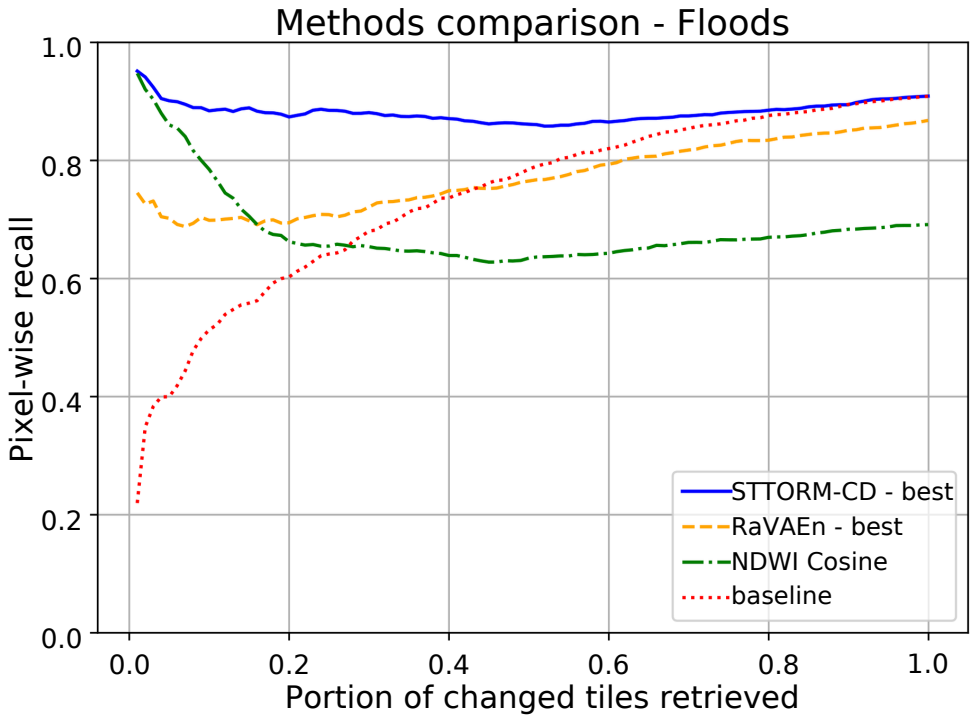

Figure 11. Methods comparison – RaVAEn-Floods dataset. The best STTORM-CD model is the small-sized one, and the best RaVAEn model is also the small-sized one. Please note that this dataset was a validation dataset for the fine-tuned model. All presented methods derived their score using $min(memory)$.

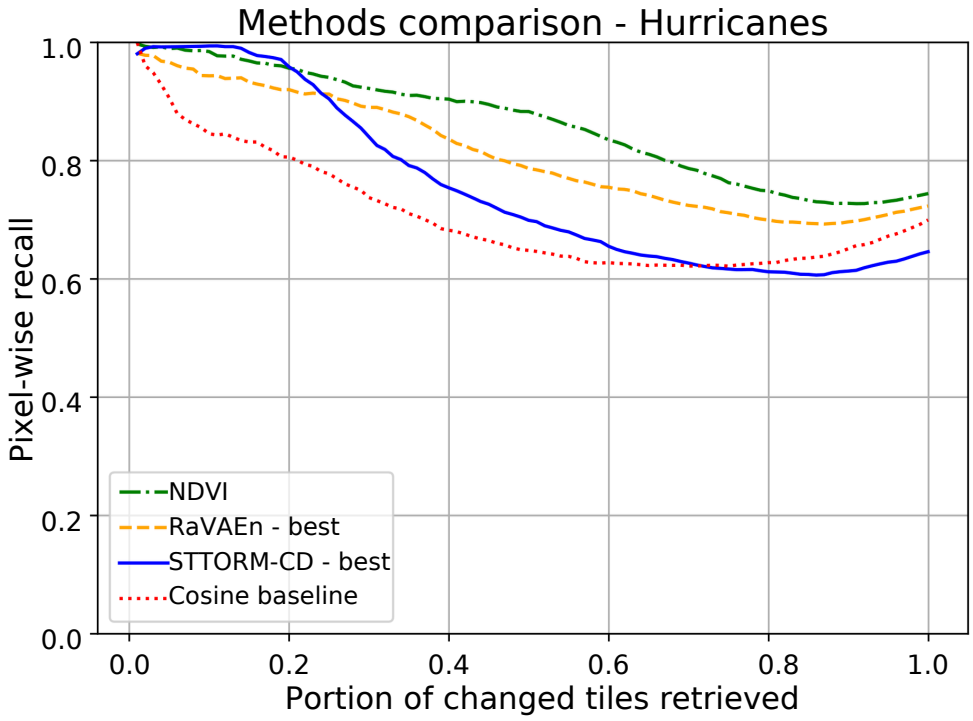

Figure 12. Methods comparison – RaVAEn-Hurricanes dataset. The best STTORM-CD model is the large-sized one, and the best RaVAEn model is the small-sized one. All presented methods derived their score using $min(memory)$.

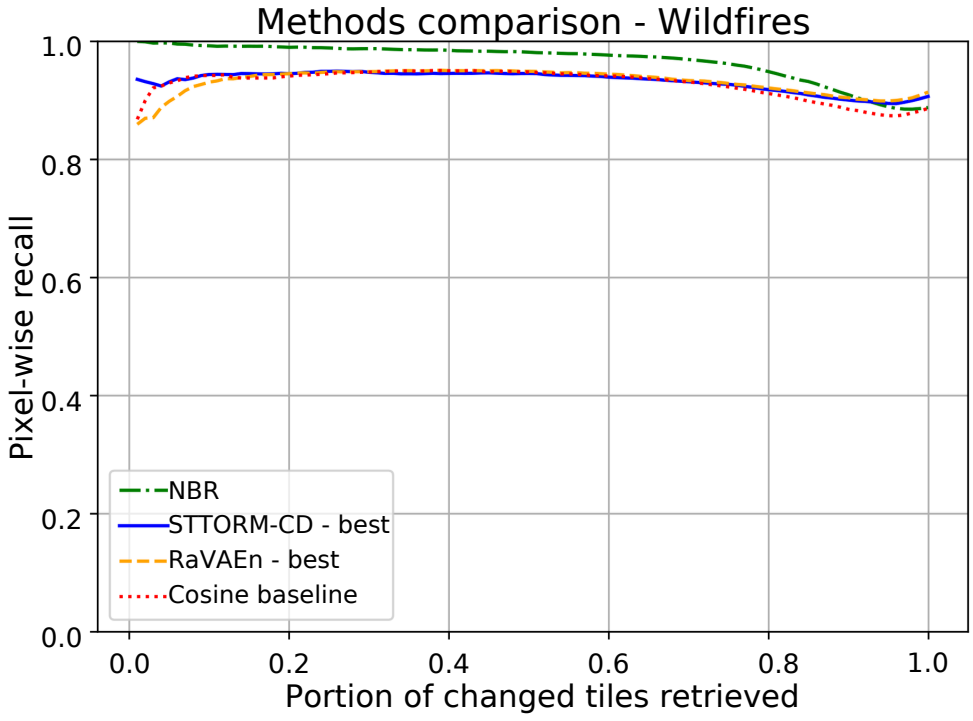

Figure 13. Methods comparison – RaVAEn-Wildfires dataset. The best STTORM-CD model is the large-sized one, and the best RaVAEn model is also the large-sized one. All presented methods derived their score using $min(memory)$.

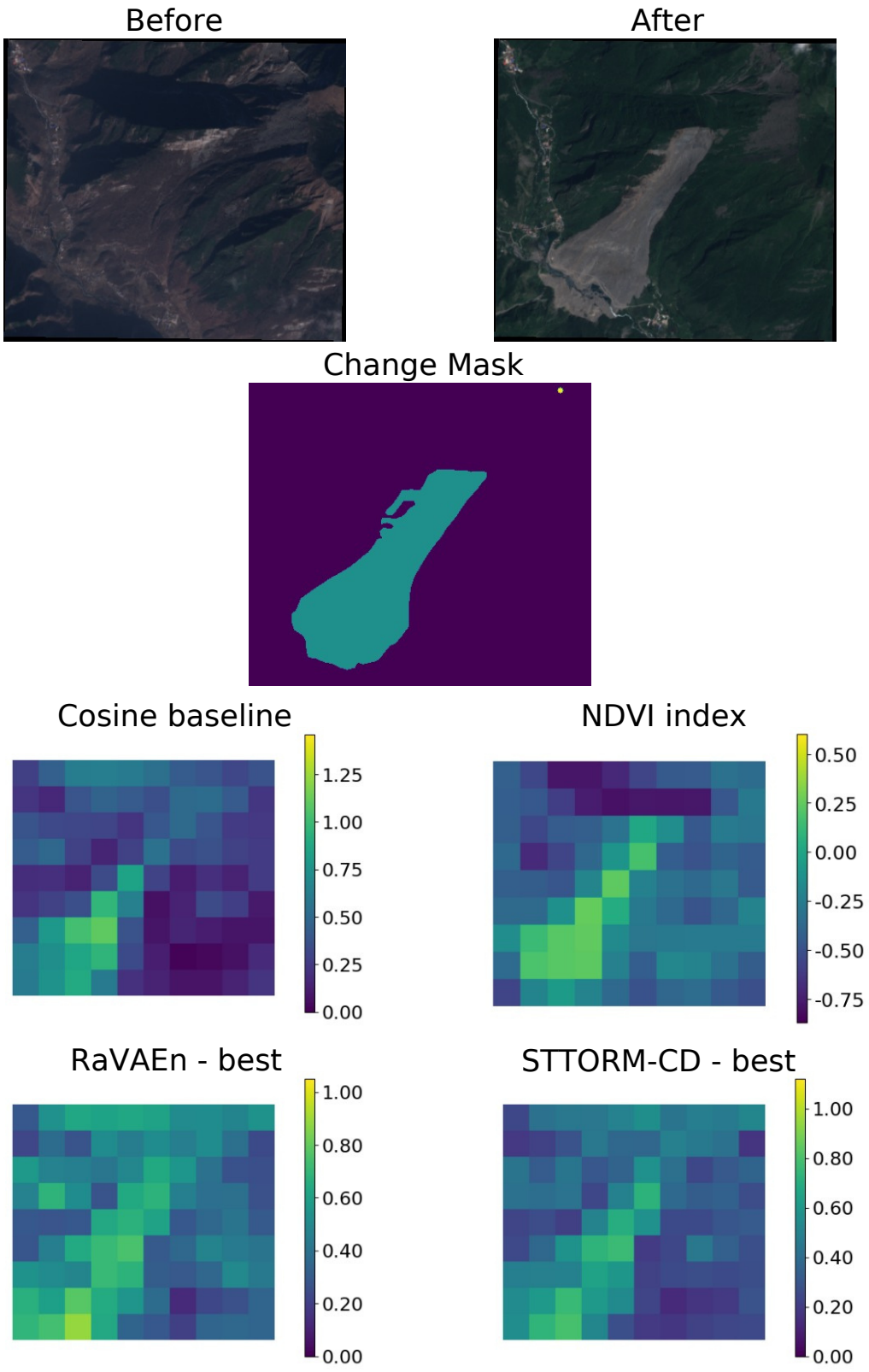

Figure 14. RaVAEn-Landslides dataset, China – Visualization of method predictions. The best STTORM-CD model is the large-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $min(memory)$.

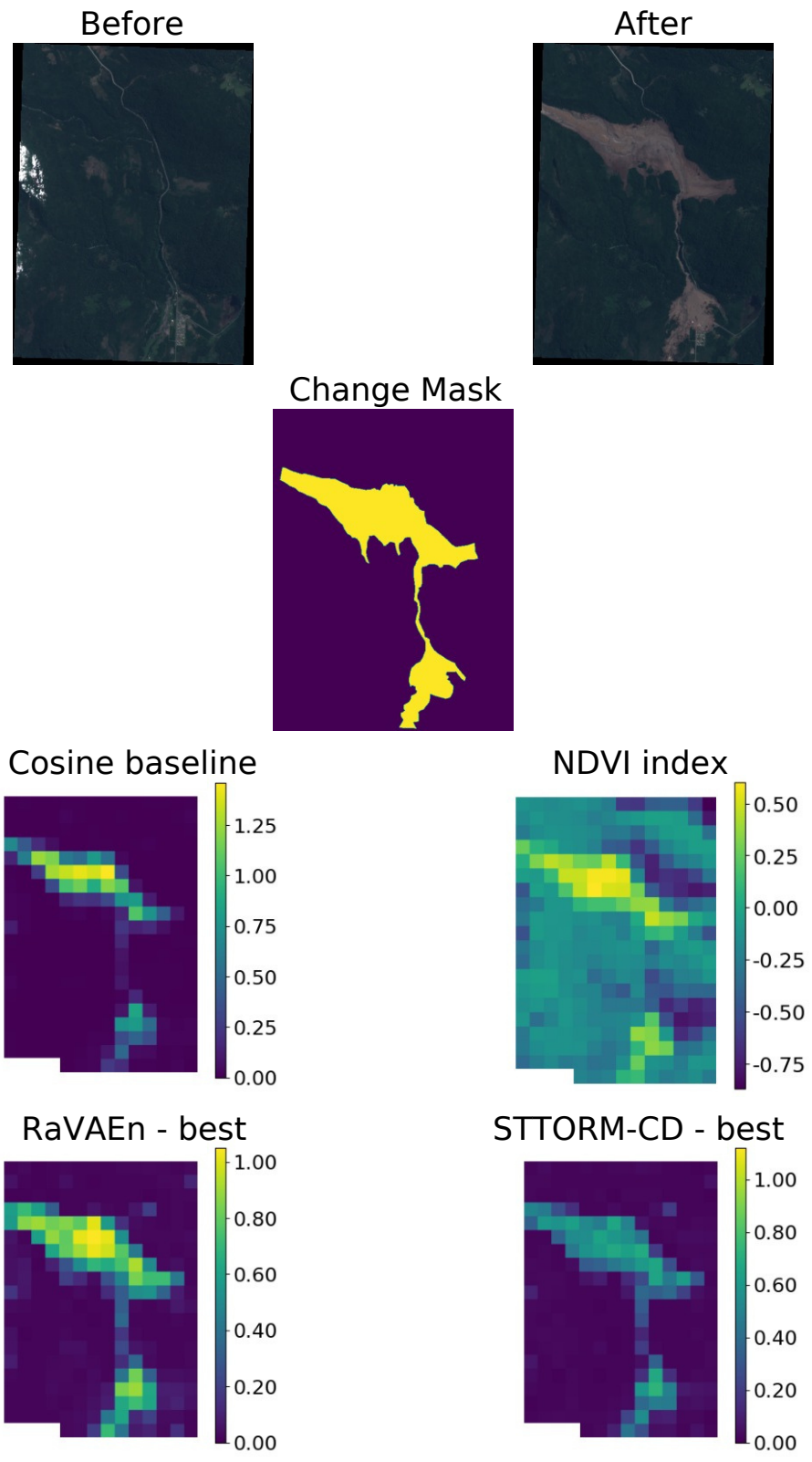

Figure 15. RaVAEn-Landslides dataset, Italy – Visualization of method predictions. The best STTORM-CD model is the large-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $min(memory)$.

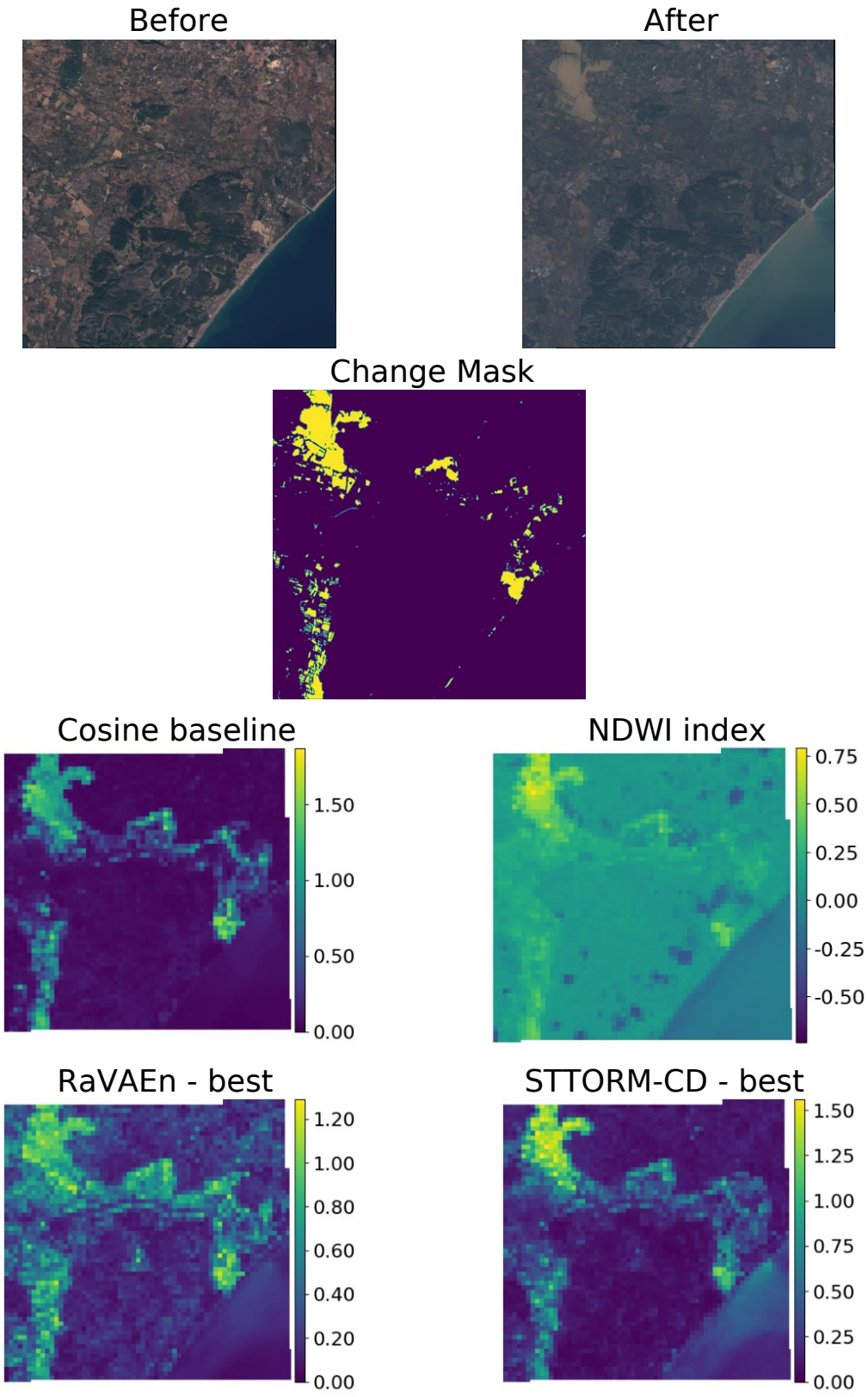

Figure 16. RaVAEn-Floods dataset, France – Visualization of method predictions. The best STTORM-CD model is the small-sized one, and the best RaVAEn model is the small-sized one. All presented methods derived their score using $min(memory)$.

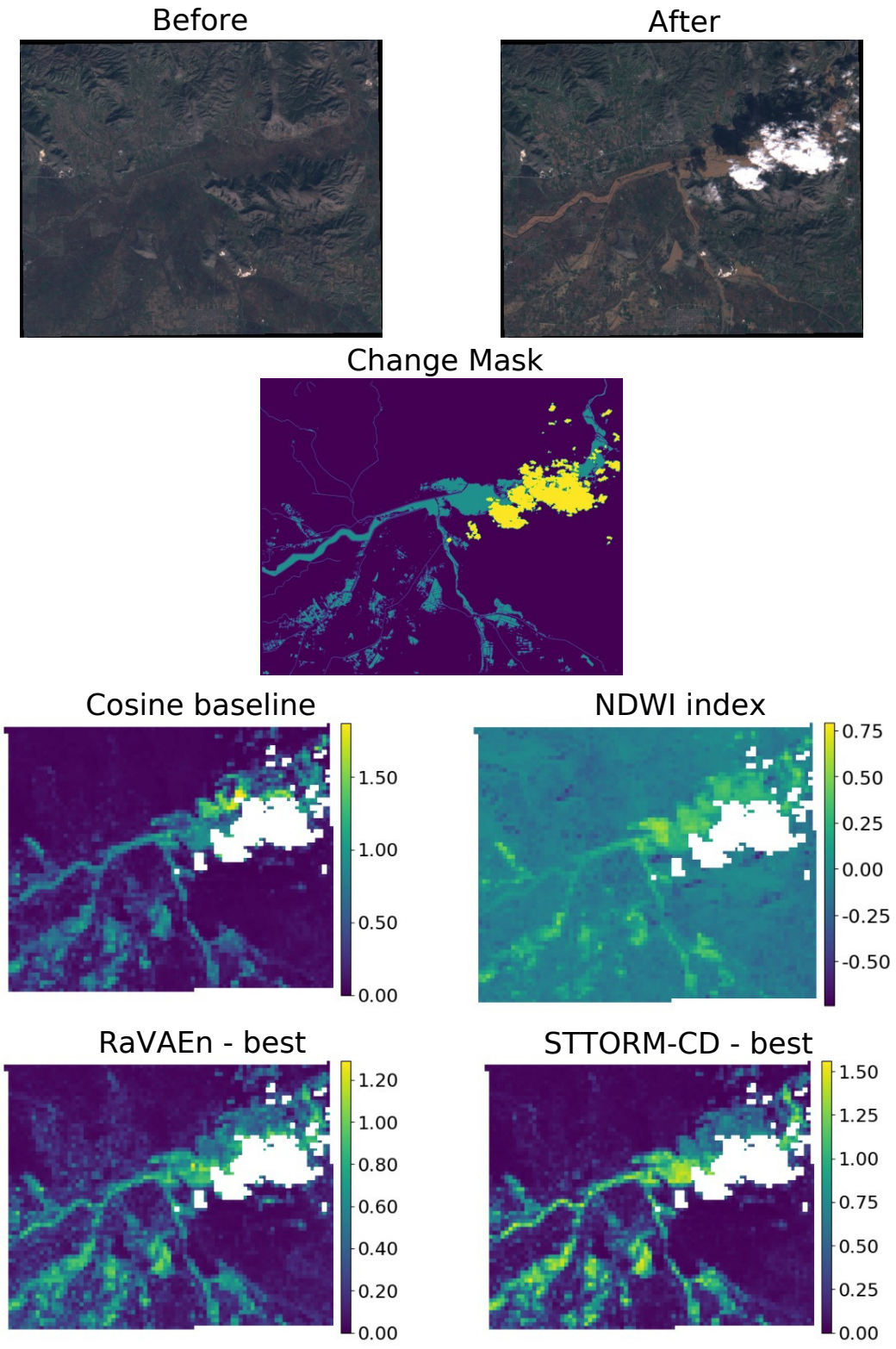

Figure 17. RaVAEn-Floods dataset, Greece – Visualization of method predictions. The best STTORM-CD model is the small-sized one, and the best RaVAEn model is the small-sized one. All presented methods derived their score using $min(memory)$.

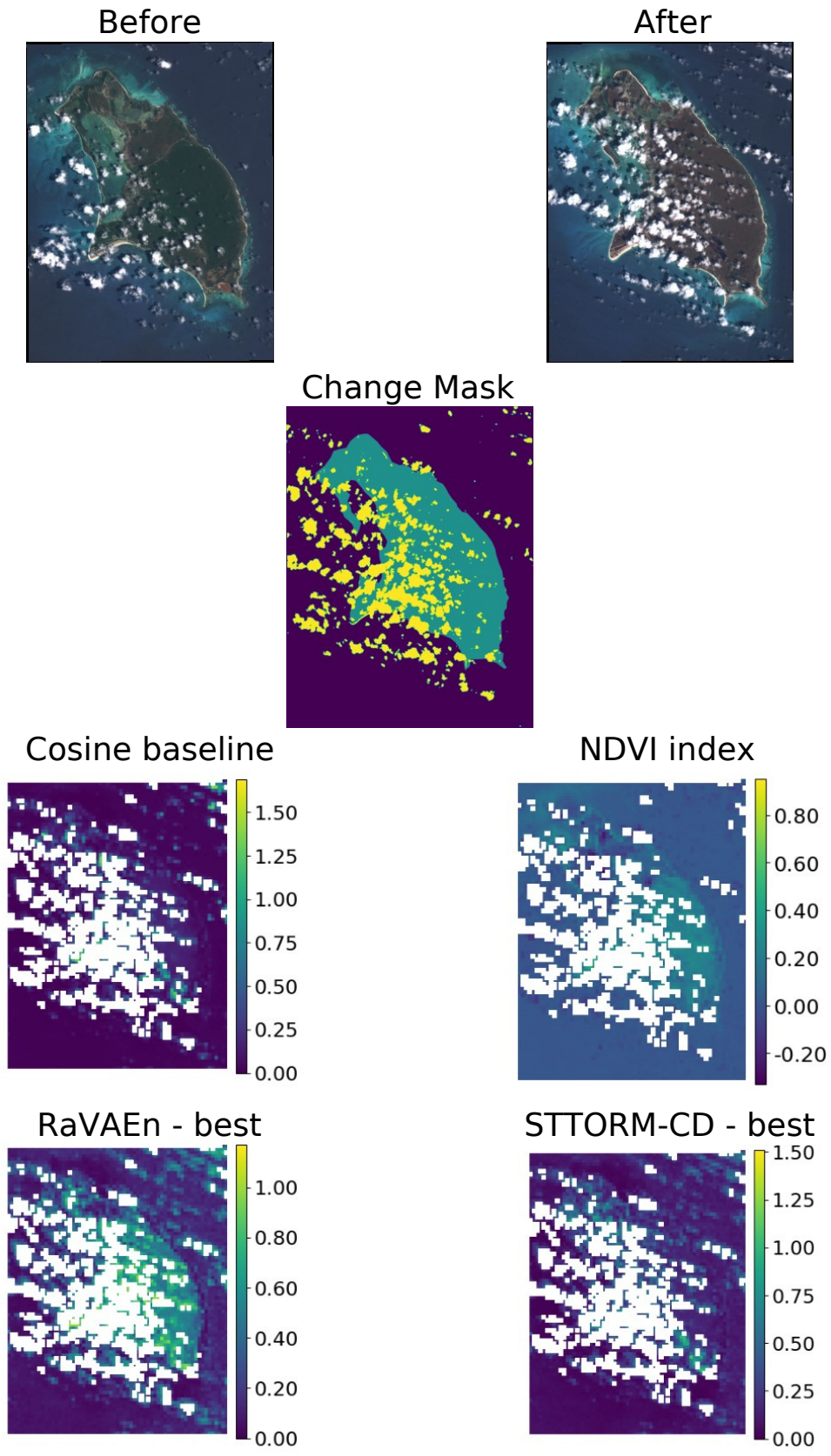

Figure 18. RaVAEn-Hurricanes dataset, Barbuda – Visualization of method predictions. The best STTORM-CD model is the large-sized one, and the best RaVAEn model is the small-sized one. All presented methods derived their score using $min(memory)$.

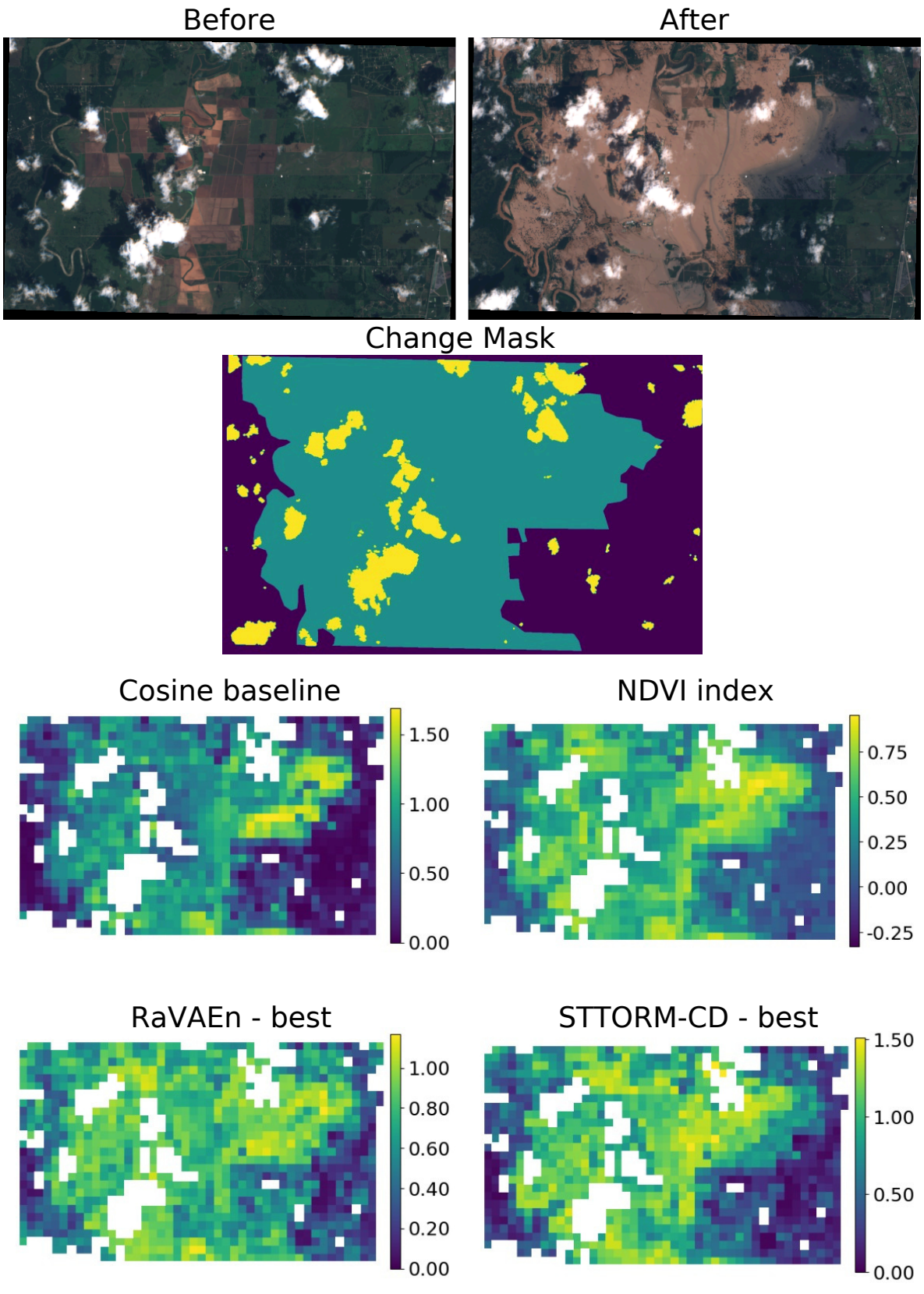

Figure 19. RaVAEn-Hurricanes dataset, USA – Visualization of method predictions. The best STTORM-CD model is the large-sized one, and the best RaVAEn model is the small-sized one. All presented methods derived their score using $min(memory)$.

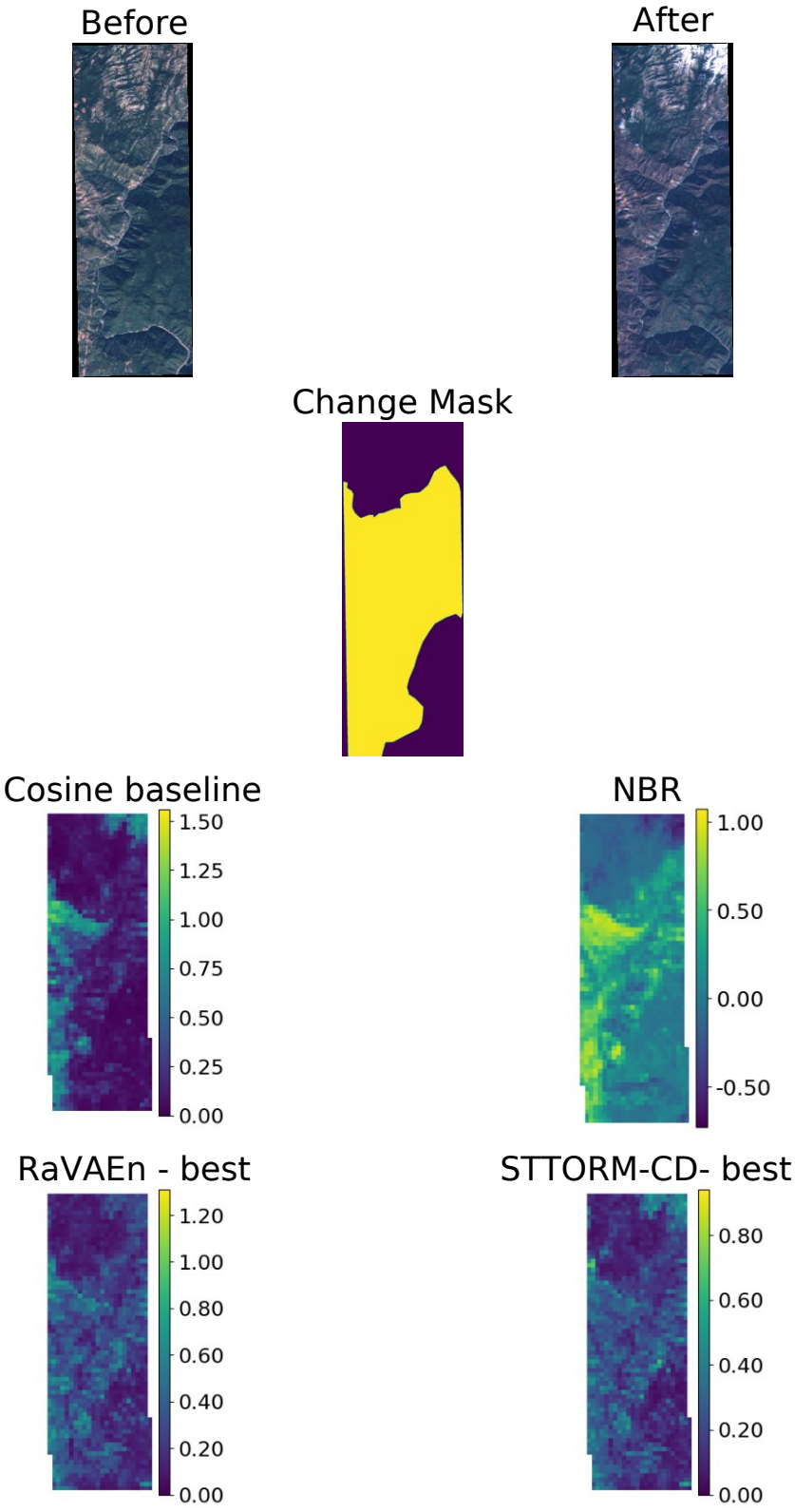

Figure 20. RaVAEn-Wildfires dataset, USA – Visualization of method predictions. The best STTORM-CD model is the large-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $min(memory)$.

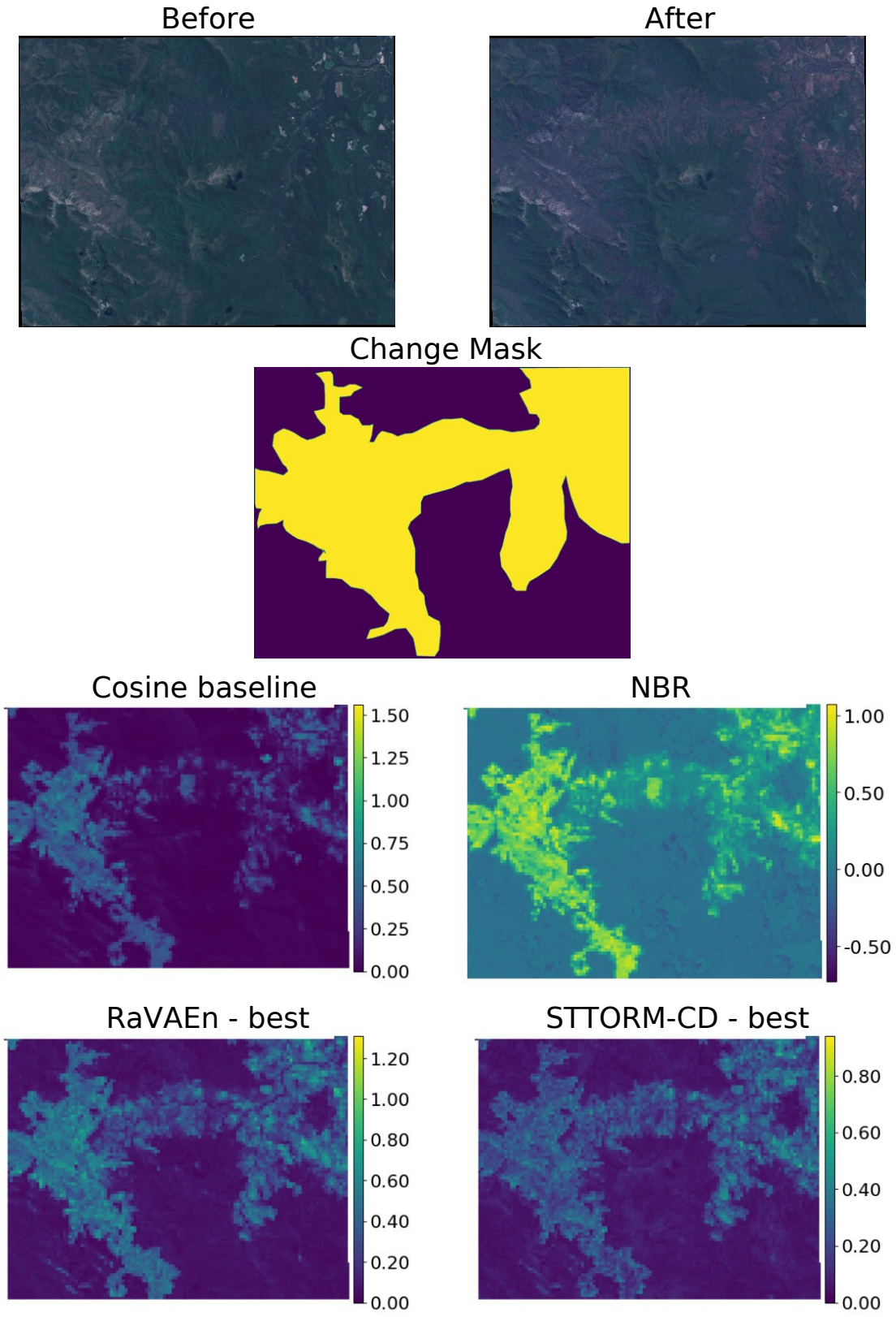

Figure 21. RaVAEn-Wildfires dataset, Australia – Visualization of method predictions. The best STTORM-CD model is the large-sized one, and the best RaVAEn model is the medium-sized one. All presented methods derived their score using $min(memory)$.