# Align and Segment: Unsupervised Learning for Building Segmentation From Misaligned Labels

Venkanna Babu Guthula[1], Oswin Krause[1], Dimitri Gominski[1], Hui Zhang[1], Johan Mottelson[2], Ankit Kariryaa[1], Nico Lang[1], and Christian Igel[1]

[1] University of Copenhagen, Copenhagen, Denmark
[2] Royal Danish Academy, Copenhagen, Denmark
{vegu, igel}@di.ku.dk

## A Appendix

In the following, we provide more details about model architecture (Section A.1), how we prepared our synthetic datasets (Section A.2) and how we prepared our real OpenStreetMap datasets (Section A.3). We also provided some qualitative and quantitative results in the respective sections.

### A.1 Details of the model architectures

***Encoder of SNet.*** In place of the SNet encoder, we experimented with three different encoders, such as ResNet-34 [3], ConvNeXt-Tiny [5] and ViT-Small [1]. We compared the three encoder backbones and compared random weight initialization to pretrained weights from ImageNet [7] for ResNet-34 and DINOv3 [8] weights for ConvNeXt-Tiny and ViT-Small. We further evaluated the impact of freezing or fine-tuning the initialized weights in encoders during training (See Tab. 1).

**Table 1: Freezing vs. training the segmentation encoders.** Results for different encoders used in the SNet. We experiment with different pretrained encoders as well as with different training strategies that either freeze or train (finetune) the encoders jointly with the TNet. These are our initial experimental results that were trained when we used a $c$ parameter of 0.2 for the output prediction range.

| | | $\mathcal{D}_{\text{uni}}$ | | | | $\mathcal{D}_{\text{bias}}$ | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Freeze | | Train | | Freeze | | Train | |
| Encoder | Weights | $\text{IoU}_{\text{seg}}$ | $\text{IoU}_{\text{align}}$ | $\text{IoU}_{\text{seg}}$ | $\text{IoU}_{\text{align}}$ | $\text{IoU}_{\text{seg}}$ | $\text{IoU}_{\text{align}}$ | $\text{IoU}_{\text{seg}}$ | $\text{IoU}_{\text{align}}$ |
| ResNet-34 | Random | 0.60 | 0.72 | 0.68 | 0.71 | 0.61 | 0.75 | 0.71 | 0.77 |
| ResNet-34 | ImageNet | 0.80 | 0.86 | 0.78 | 0.80 | 0.68 | 0.72 | 0.41 | 0.43 |
| ConvNeXt | DINOv3 | 0.81 | 0.85 | 0.72 | 0.72 | 0.69 | 0.72 | 0.41 | 0.43 |
| ViT-Small | DINOv3 | 0.79 | 0.84 | 0.72 | 0.73 | 0.70 | 0.73 | 0.40 | 0.41 |

The results of our comparison of model architectures and initialization choices are shown in Tab. 1. For all weight initializations (except random) and all datasets, freezing the pre-initialized weights outperformed fine-tuning. Indeed, we can see that for the frozen weights, the networks showed reduced bias, which is reflected in the larger $\text{IoU}_{\text{seg}}$ and $\text{IoU}_{\text{align}}$ compared to trained variants. This effect was more pronounced on $\mathcal{D}_{\text{bias}}$ than on $\mathcal{D}_{\text{uni}}$. Finally, pre-trained initializations consistently outperformed random initialization in this study; overall, the ConvNeXt backbone with pretrained weights from DINOv3 performed best.

***Decoder of SNet.*** When using ResNet-34 as an encoder, we used the standard U-Net decoder. The decoder for ConvNeXt-Tiny and ViT-Small was composed of convolutional and bilinear upsampling layers [6], where each convolution was followed by a GeLU activation function [4]. Based on the backbone, we adapted the number of skip connections from the different encoder stages. All variants employed skip connections at the input layer at full resolution, implemented using two convolutional layers of kernel size $1 \times 1$. The ResNet-34, ConvNeXt-Tiny and ViT-Small models used additional three, two, and no skip-connections, respectively, each with a convolution using a $3 \times 3$ kernel.

### A.2 Synthetic Dataset

**Data preparation.** We created misaligned datasets for three cities in the SpaceNet-2 [2] benchmark by applying affine transformations to the labels in every patch. We created patches of $320 \times 320$ pixels from the original $650 \times 650$ patches without any overlap to predict misalignment for a smaller region. To generate the dataset $\mathcal{D}_{\text{bias}}$, we apply the same misalignment for every patch, i.e., 50 pixels translation on the x-axis (see last image in Fig 1). We used this transformation in all but the robustness experiment.

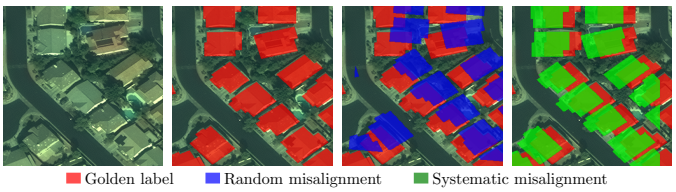


**Fig. 1:** Examples of $I_x$ (input image), $I_y$ (golden label) and $I_{y'}$ (artificially misaligned labels). In the third column, $I_{y'}$ was generated by assigning random misalignment with magnitude of $\|\mathcal{A}_\theta - Id\|_F = 0.241$, $t_x = -48$, $t_y = 49$, and $r \approx 4.5°$. In the fourth column, $I_{y'}$ was generated using systematic misalignment with magnitude of $\|\mathcal{A}_\theta - Id\|_F = 0.156$, $t_x = 50$, $t_y = 0$, and $r \approx 0°$.

To generate the random dataset $\mathcal{D}_{\text{uni}}$, we assigned a random misalignment to every patch. This random misalignment depends on a single parameter $s_{\max}$

limiting the *maximum shift*. When we generate random misalignment, we need the translation parameters $t_x$, $t_y$ in pixels and rotation angle $\alpha$ in radians to create an affine transformation $\mathcal{A}_\theta$. We sample $N$ sets of parameters, by sampling $t_x, t_y$ uniformly in $[s_{\max}, s_{\max}]$ and $\alpha$ uniformly in $[-\frac{s_{\max}}{2 \cdot 320}, \frac{s_{\max}}{2 \cdot 320}]$. As a result, with $s_{\max} = 50$, the range of the resulting rotations is $[-4.5, 4.5]$ degrees. See Fig. 1 for an example from $\mathcal{D}_{\text{uni}}$ and $\mathcal{D}_{\text{bias}}$.

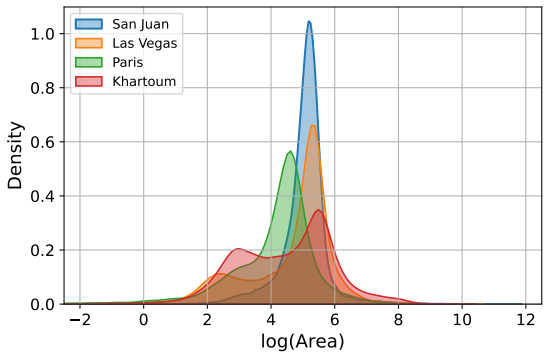


**Fig. 2: Distribution of area of buildings in all cities.** The area of buildings is measured in square meters before applying a log scale on the x-axis. The size of buildings varies across cities. For example, it is high in Las Vegas and San Juan and concentrated, and it is smaller in Paris compared to Las Vegas and San Juan. Buildings in Khartoum are distributed with two peaks. The size of buildings in all these cities can be observed in figures that show predictions (from Fig 3 to Fig 8).

**Predictions.** Since the alignment and segmentation performance also depends on the size of buildings, the distribution of building size is visualized in Fig. 2. Compared to other cities, the size of buildings is bigger in Las Vegas, and this leads to good overlap between original buildings (or golden labels) and misaligned labels. This can be a good reason why the performance in this city is better compared to the other two cities. While the alignment masks corrected very well with better $\text{IoU}_{\text{align}}$, the segmentation masks required some improvements. We presented some examples of images and predictions in each city. Fig. 3 and Fig. 4 shows examples from $\mathcal{D}_{\text{uni}}$ and $\mathcal{D}_{\text{bias}}$, respectively, from Las Vegas. Fig. 5 and Fig. 6 shows examples from $\mathcal{D}_{\text{uni}}$ and $\mathcal{D}_{\text{bias}}$, respectively, from Paris. The buildings are smaller compared to Las Vegas, leading to less or no overlap between golden labels and misaligned masks. This can be the reason for low scores of both $\text{IoU}_{\text{seg}}$ and $\text{IoU}_{\text{align}}$. Fig. 7 and Fig. 8 shows examples from $\mathcal{D}_{\text{uni}}$ and $\mathcal{D}_{\text{bias}}$, respectively, from Khartoum city. The sizes of buildings in the city range from small to large.

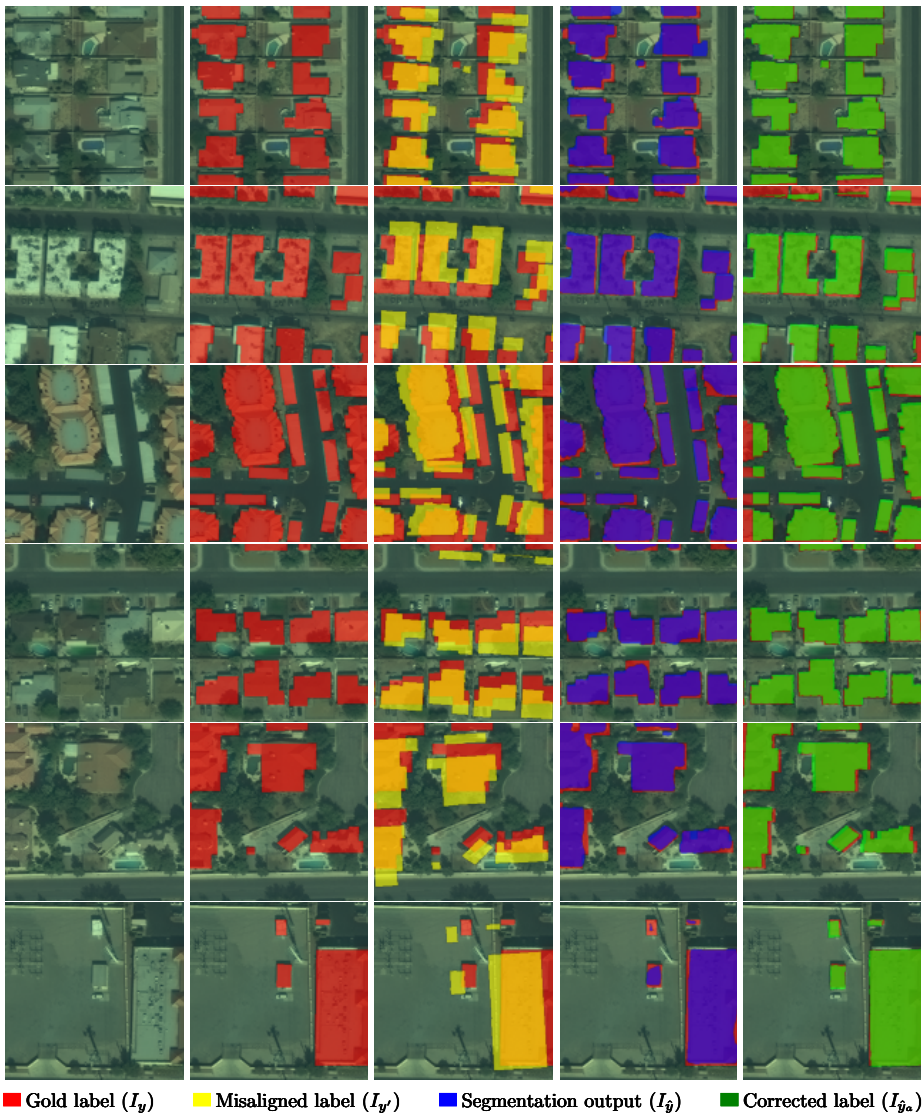


**Fig. 3: Example images and predictions from $\mathcal{D}_{\text{uni}}$ of Las Vegas**. The first column shows the RGB image ($I_x$) and the second column shows the golden labels ($I_y$). The third column presents the misalignment by comparing the misaligned labels ($I_{y'}$) with $I_y$. The fourth and fifth columns present predicted segmentation output ($I_{\hat{y}}$) and corrected label ($I_{\hat{y}_a}$) quality by comparing both with $I_y$.

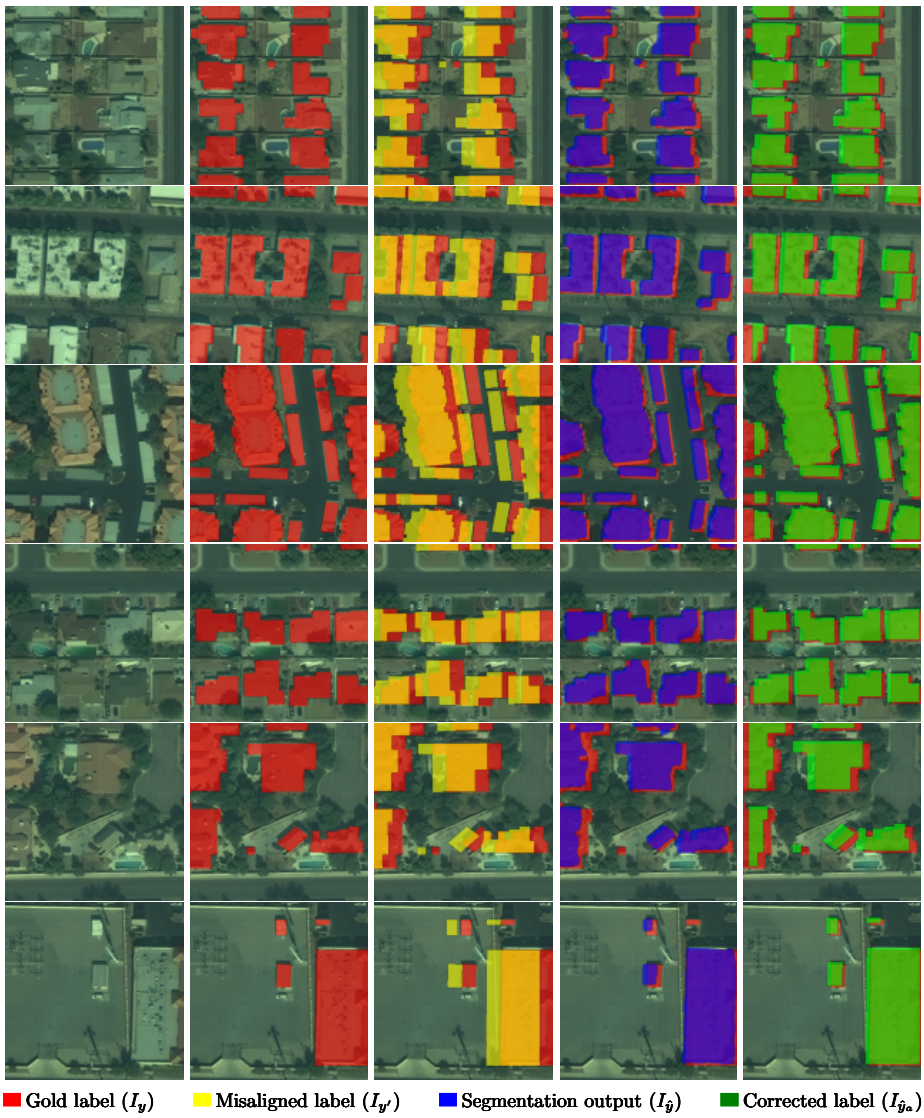


**Fig. 4: Example images and predictions from $\mathcal{D}_{\text{bias}}$ of Las Vegas.** The first column shows the RGB image ($I_x$) and the second column shows the golden labels ($I_y$). The third column presents the misalignment by comparing the misaligned labels ($I_{y'}$) with $I_y$. The fourth and fifth columns present predicted segmentation output ($I_{\hat{y}}$) and corrected label ($I_{\hat{y}_a}$) quality by comparing both with $I_y$.

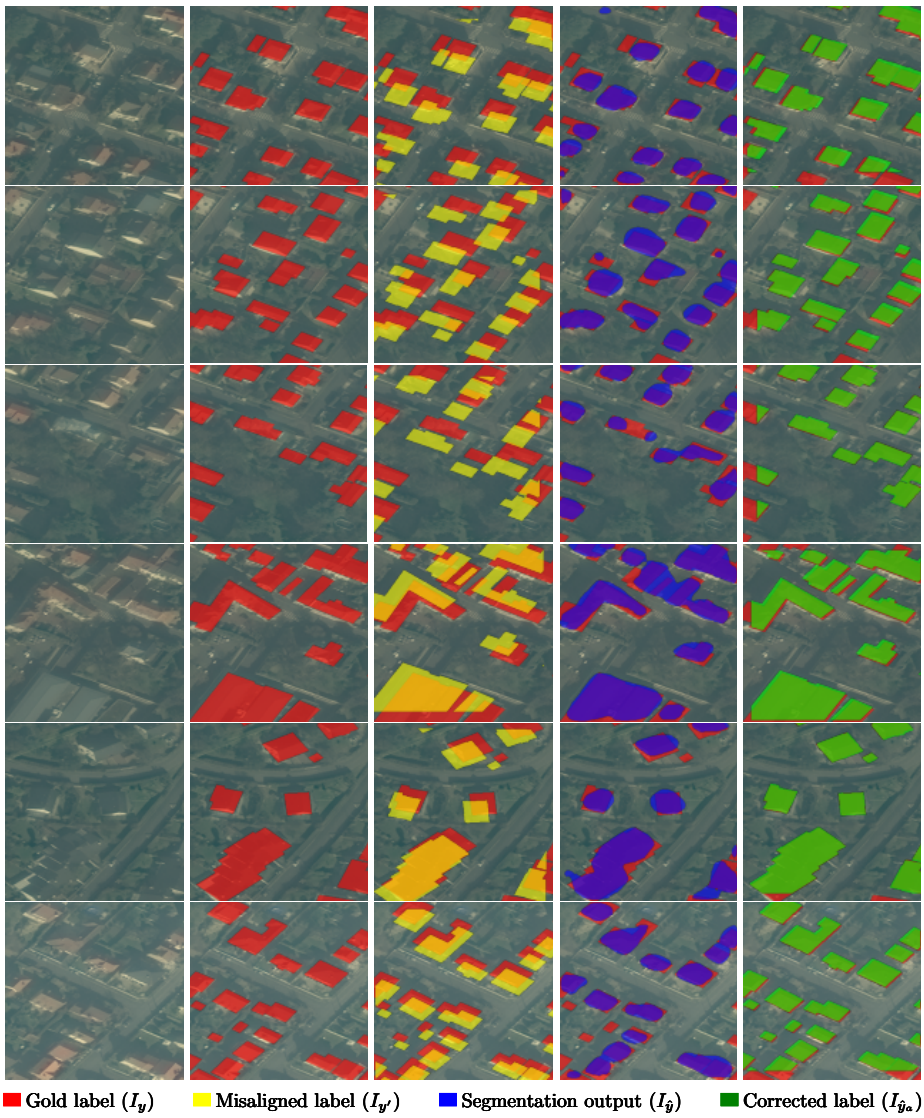


**Fig. 5: Example images and predictions from $\mathcal{D}_{\text{uni}}$ of Paris** The first column shows the RGB image ($I_x$) and the second column shows the golden labels ($I_y$). The third column presents the misalignment by comparing the misaligned labels ($I_{y'}$) with $I_y$. The fourth and fifth columns present predicted segmentation output ($I_{\hat{y}}$) and corrected label ($I_{\hat{y}_a}$) quality by comparing both with $I_y$.

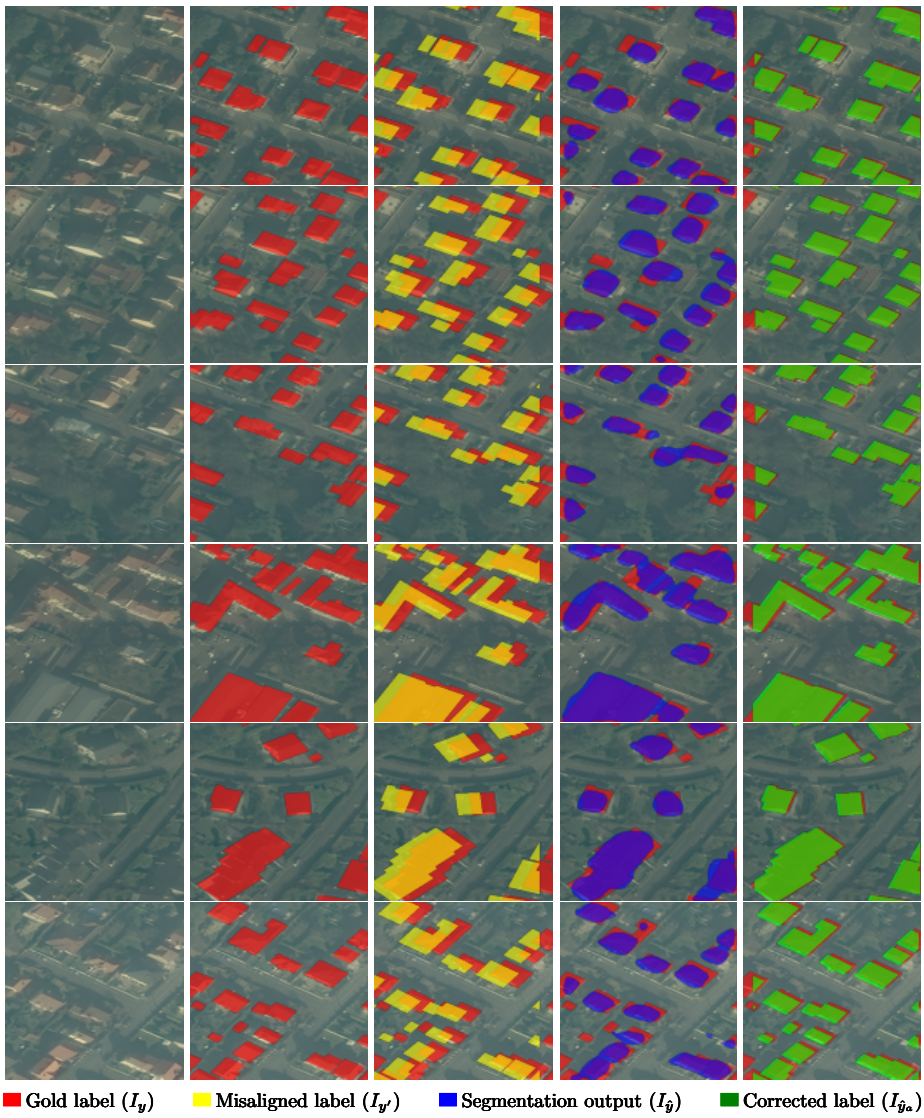


**Fig. 6: Example images and predictions from $\mathcal{D}_{\text{bias}}$ of Paris**. The first column shows the RGB image ($I_x$) and the second column shows the golden labels ($I_y$). The third column presents the misalignment by comparing the misaligned labels ($I_{y'}$) with $I_y$. The fourth and fifth columns present predicted segmentation output ($I_{\hat{y}}$) and corrected label ($I_{\hat{y}_a}$) quality by comparing both with $I_y$.

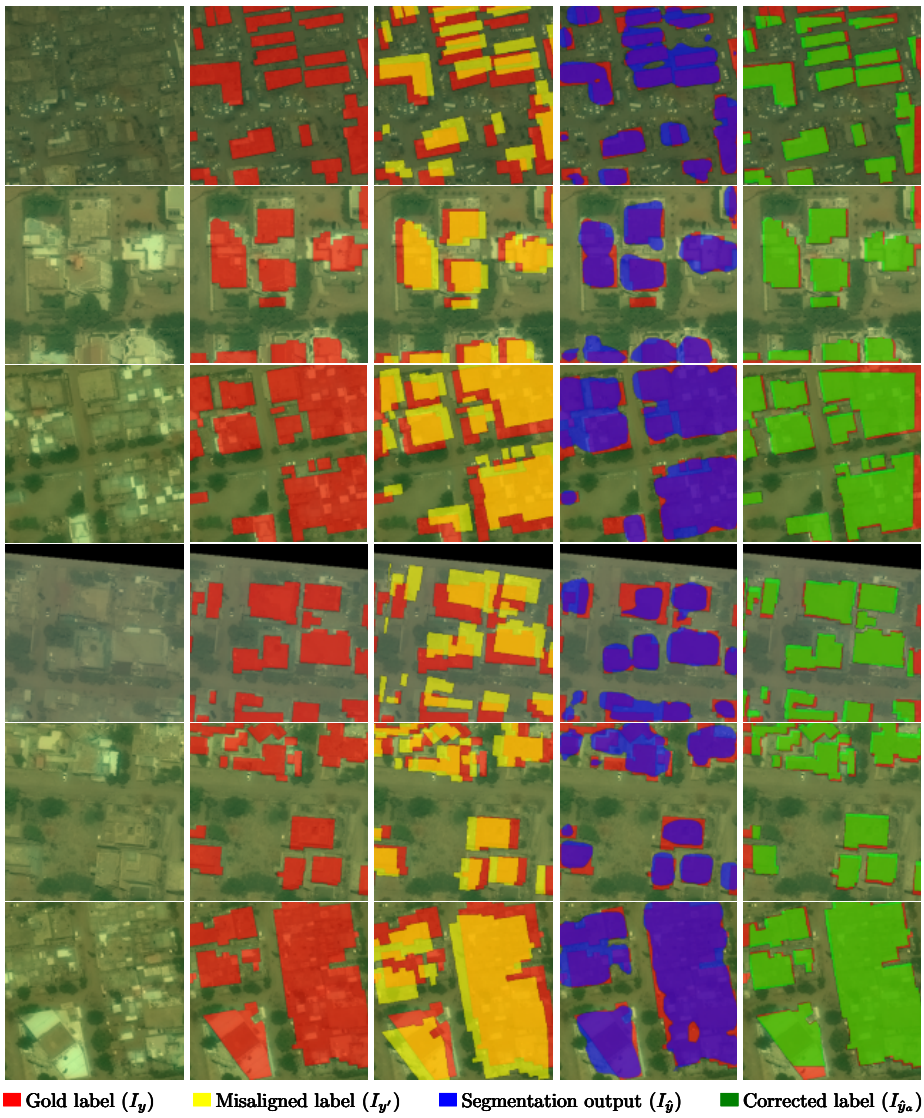


**Fig. 7: Example images and predictions from $\mathcal{D}_{\text{uni}}$ of Khartoum.** The first column shows the RGB image ($I_x$) and the second column shows the golden labels ($I_y$). The third column presents the misalignment by comparing the misaligned labels ($I_{y'}$) with $I_y$. The fourth and fifth columns present predicted segmentation output ($I_{\hat{y}}$) and corrected label ($I_{\hat{y}_a}$) quality by comparing both with $I_y$.

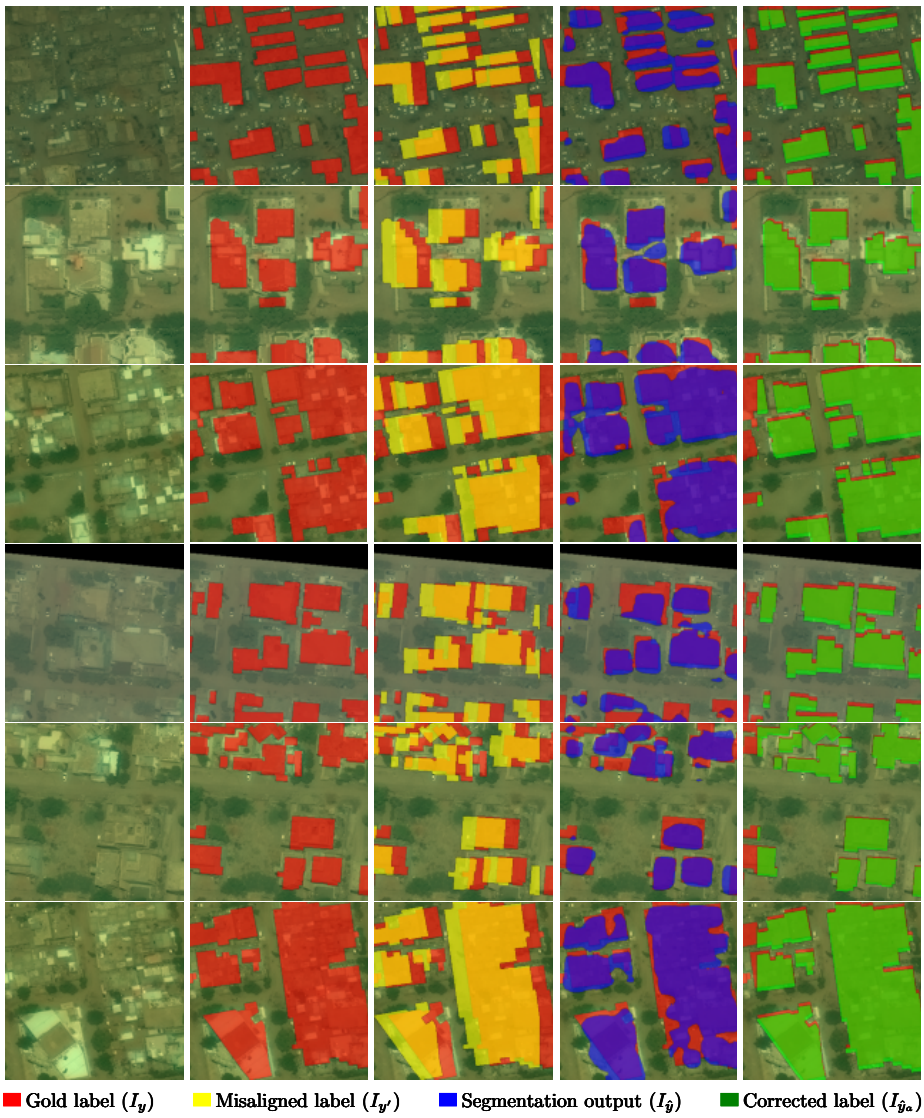


**Fig. 8: Example images and predictions from $\mathcal{D}_{\text{bias}}$ of Khartoum.** The first column shows the RGB image ($I_x$) and the second column shows the golden labels ($I_y$). The third column presents the misalignment by comparing the misaligned labels ($I_{y'}$) with $I_y$. The fourth and fifth columns present predicted segmentation output ($I_{\hat{y}}$) and corrected label ($I_{\hat{y}_a}$) quality by comparing both with $I_y$.

### A.3 OpenStreetMap dataset

**Data preparation.** Fig. 9 shows the imagery and building footprints we used as a part of qualitative analysis on a real OpenStreetMap dataset. We obtained original imagery from SpaceNet-5 [2] and building footprints from OpenStreetMap. Then the entire imagery divided into small patches of $320 \times 320$ pixels without any overlap. Patches without building footprints are removed, and the remaining patches are split into training, validation, and test sets in an 8:1:1 ratio.

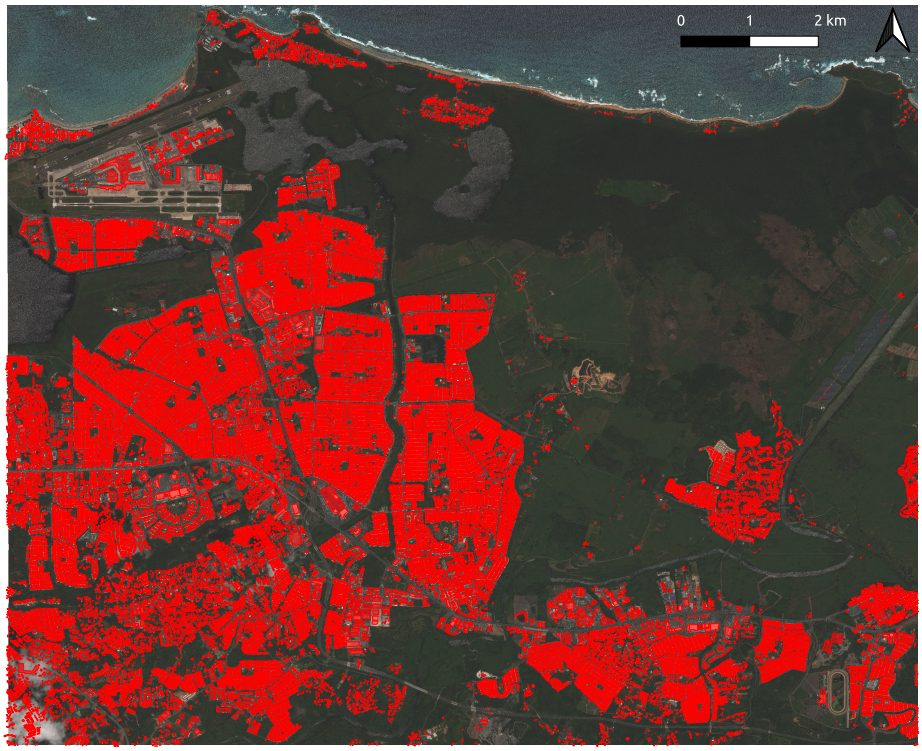


**Fig. 9: Imagery and OpenStreetMap building footprints of San Juan city.**

**Predictions.** Fig. 10 shows more qualitative examples of predictions and Fig. 11 shows the translation of the predicted affine transformation of every patch as a flow field. The flow field only prepared using translations $t_x$ and $t_y$ because of predictions of a very small rotation. The color represents the magnitude of the predicted transformation that estimated the Frobenius norm of $\mathcal{A}_\theta^{-1}$ after subtracting an identity matrix from it.

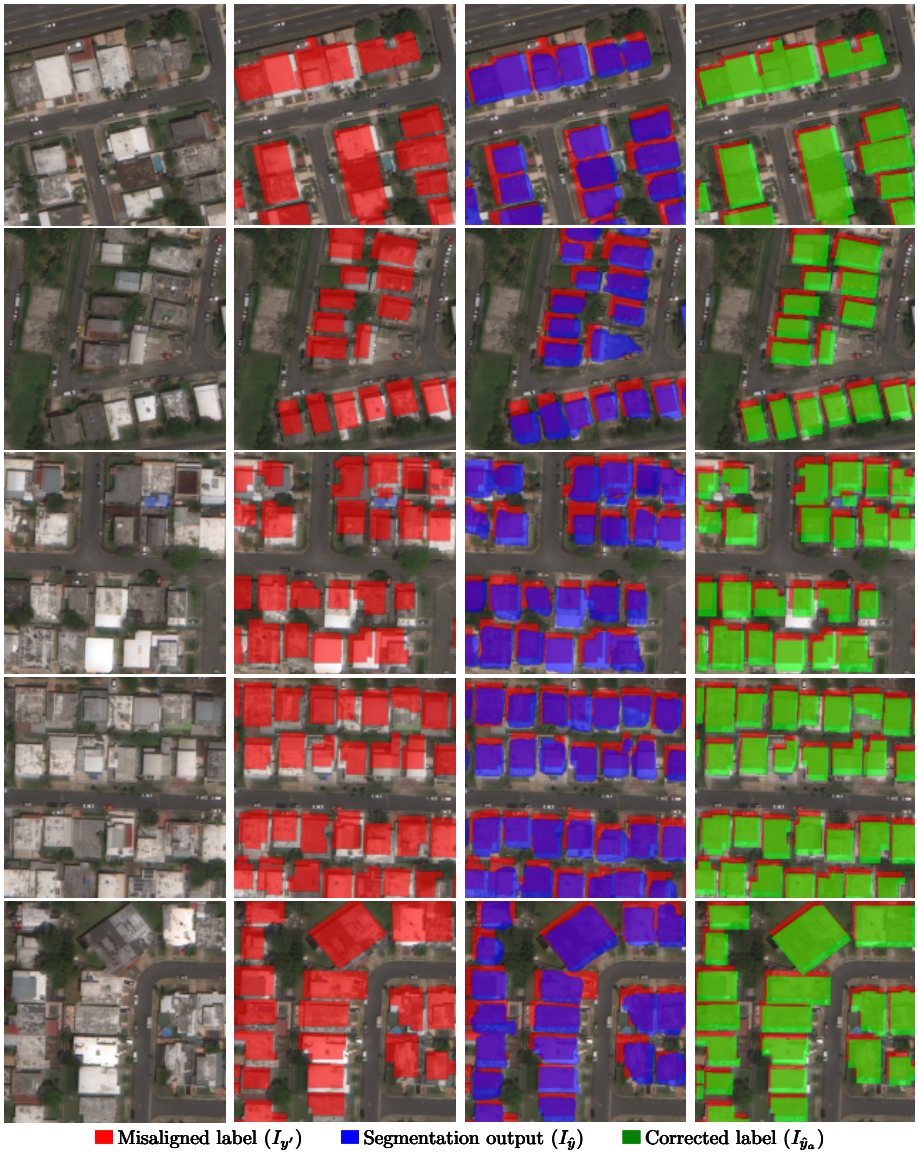


**Fig. 10: Example images and predictions from San Juan city data.** The first column shows the RGB image ($I_x$) and the second column shows the misaligned labels ($I_{y'}$) from OpenStreetMap data. The third and fourth columns present predicted segmentation output ($I_{\hat{y}}$) and corrected label ($I_{\hat{y}_a}$) quality by comparing both with $I_{y'}$.

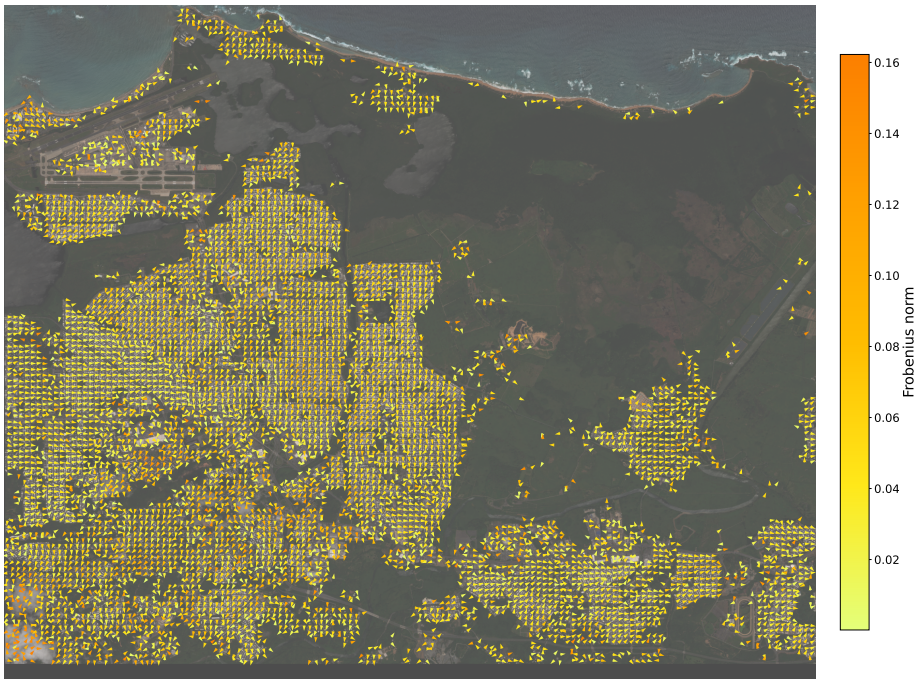


**Fig. 11: The predicted transformation over the entire San Juan city.** The flow field was generated using the translations predicted using the model. The colorbar shows the magnitude of the predicted transformation.

## References


1. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (ICLR) (2021)
2. Etten, A.V., Lindenbaum, D., Bacastow, T.M.: SpaceNet: A remote sensing dataset and challenge series (2019)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
4. Hendrycks, D., Gimpel, K.: Gaussian error linear units (GELUs) (2016)
5. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A ConvNet for the 2020s. In: Computer Vision and Pattern Recognition (CVPR). pp. 11976–11986 (2022)
6. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer Assisted Intervention (MICCAI). pp. 234–241. Springer (2015)
7. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015)
8. Siméoni, O., Vo, H.V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., Yi, S., Ramamonjisoa, M., Massa, F., Haziza, D., Wehrstedt, L., Wang, J., Darcet, T., Moutakanni, T., Sentana, L., Roberts, C., Vedaldi, A., Tolan, J., Brandt, J., Couprie, C., Mairal, J., Jégou, H., Labatut, P., Bojanowski, P.: DINOv3 (2025)