Supplementary material for: "ViTs for SITS: Vision Transformers for Satellite Image Time Series"

Michail Tarasiou Erik Chavez Stefanos Zafeiriou
Imperial College London
{michail.tarasiou10, erik.chavez, s.zafeiriou}@imperial.ac.uk

1. Datasets

1.1. Training data

Normalized pixel counts for the 17 distinct classes found in the **Germany** dataset [5] are presented in Fig.1. In experiments, we use the 2016 data which consist of 27k training and 8.5k evaluation samples of size 24×24 pixels with 13 image bands. To construct the classification dataset we simply select all samples whose center pixel locations, at indices (12:13, 12:13), belong to the same class (to avoid ambiguities in sample class attribution) assuming it is not the background class. We keep training and evaluation data splits the same as the segmentation set. Finally, we discard four object classes (hop, beans, soybeans, peas) that now consist of less than 100 samples in the training data. This leaves us with 13k and 4k training and evaluation samples respectively whose class distribution is shown in Fig.1.

Equivalently, the **T31TFM-1618** dataset [6] contains 120k and 20k training and evaluation samples of size 48×48 with 13 image bands and 20 distinct classes. Per-class normalized pixel counts are presented in Fig.3. To construct the classification dataset we follow the same strategy as described above for Germany, we select only samples whose 2×2 center region is occupied by the same class, assuming it is not the background class. We further exclude two land cover types (J5M, MC7) with less than 50 samples in the training set, leaving us with 45k and 8k training and evaluation samples whose class distribution is shown in Fig.3.

The **PASTIS** dataset [1] contains 2.4k satellite image timeseries (SITS) samples of size 128×128 each with 33-61 temporal acquisitions and 10 image bands. To reduce computational requirements during experiments, we split PASTIS samples into 24×24 patches in space while retaining all acquisition times. This allows us to retain as much information as possible with respect to the original dataset; while losing acquisitions would alter the information content of data samples, model outputs can always be reassembled back into original dimensions. Besides, we have empirically found from experiments in Germany that there is no performance drop in training with small size inputs. An image size 24×24 was selected as it fits the dimension requirements of all tested models: 1) it can be used with patch sizes 2, 3, 4, 6 with the Temporo-Spatial Vision Transformer (TSViT), 2) it is a multiple of 8 which makes it suitable for the CNN-based architectures tested, e.g. a 24×24 size input to UNET3D leads to residual decoder feature maps matching the dimensions of respective encoder maps. In total, we get 60.5k samples of size 24×24 matching the original data splits. To accommodate a large set of experiments we only use fold-1 among the five folds provided in PASTIS. This corresponds to provided sets 1, 2, 3 used for training (36.5k samples), 4 for evaluation (12k samples) and 5 as a test set (12k samples). To directly compare with [1] we further train TSViT on all 5 folds. In experiments with PASTIS we treat the *background* class as another crop type to predict, while masking the effect of the *void* label from the training loss and evaluation metrics. We report the performance on the test data of the model found to perform best on the evaluation set. To make the PASTIS classification dataset we took advantage of the object instance ids provided to extract 24×24 pixel regions whose center pixel falls inside each object and use the class of this object as the sample class. As a result, the PASTIS classification dataset contains 34k training, 11.5k evaluation and 11.5k test samples and all the classes found in the segmentation set. Pixel and parcel class counts for the segmentation and classification datasets can be seen in Fig.2.

1.2. Per-class distribution of neighbours

In section 3.4 we presented our motivation for the design of the TSViT encoder, arguing that the use of spatial patterns in crop recognition is undermined by the lack of structure with respect to the relative positioning of crop types within an area of interest. We further hypothesize that the spatial distribution of crop types is independent of their proximity to other crop types over large regions, i.e. what grows in one location does not provide generalizable information about what grows in nearby locations. to test this hypothesis we use the T31TFM *S2* tile in France for years 2016 to 2018, covering a $100km \times 100km$ region. For this

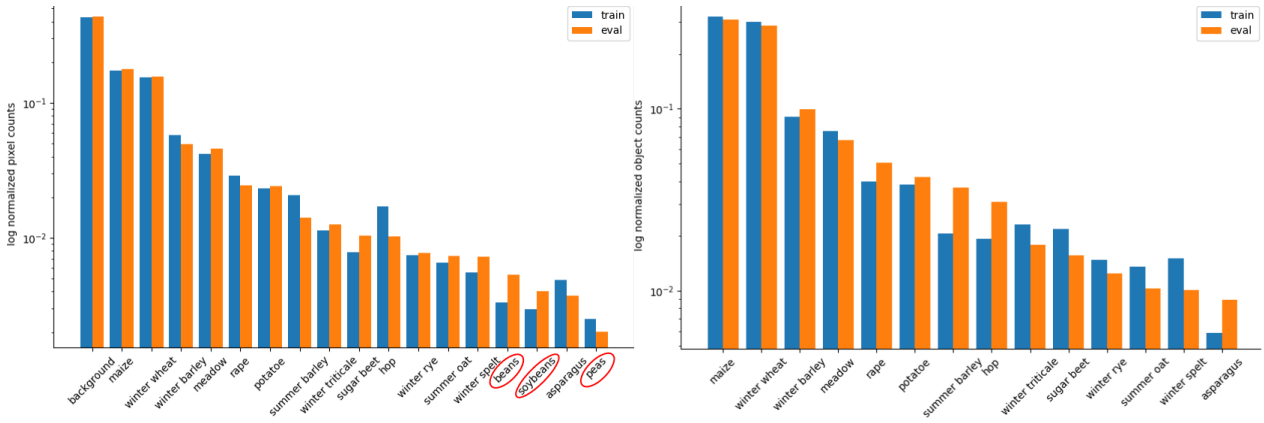

Figure 1. **Distribution of ground truth classes for the Germany dataset** [5]. **(left)** semantic segmentation data. Circled ground truths are not included in classification data. **(right)** object classification data.

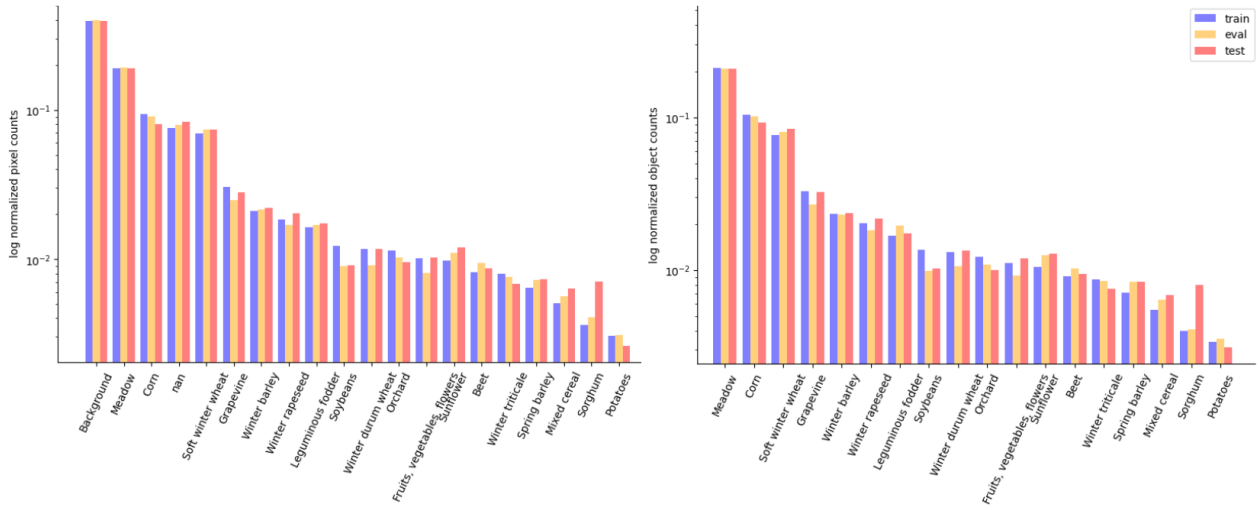

Figure 2. **Distribution of ground truth classes for the PASTIS dataset** [1]. **(left)** semantic segmentation data. **(right)** object classification data.

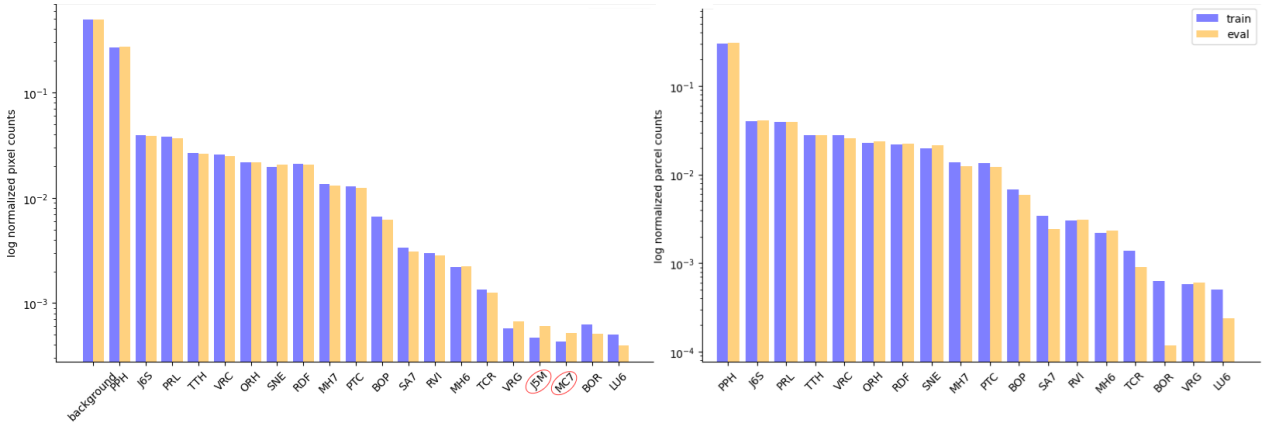

Figure 3. **Distribution of ground truth classes for the T31TFM-1618 dataset** [6]. **(left)** semantic segmentation data. Circled ground truths are not included in classification data. **(right)** object classification data.

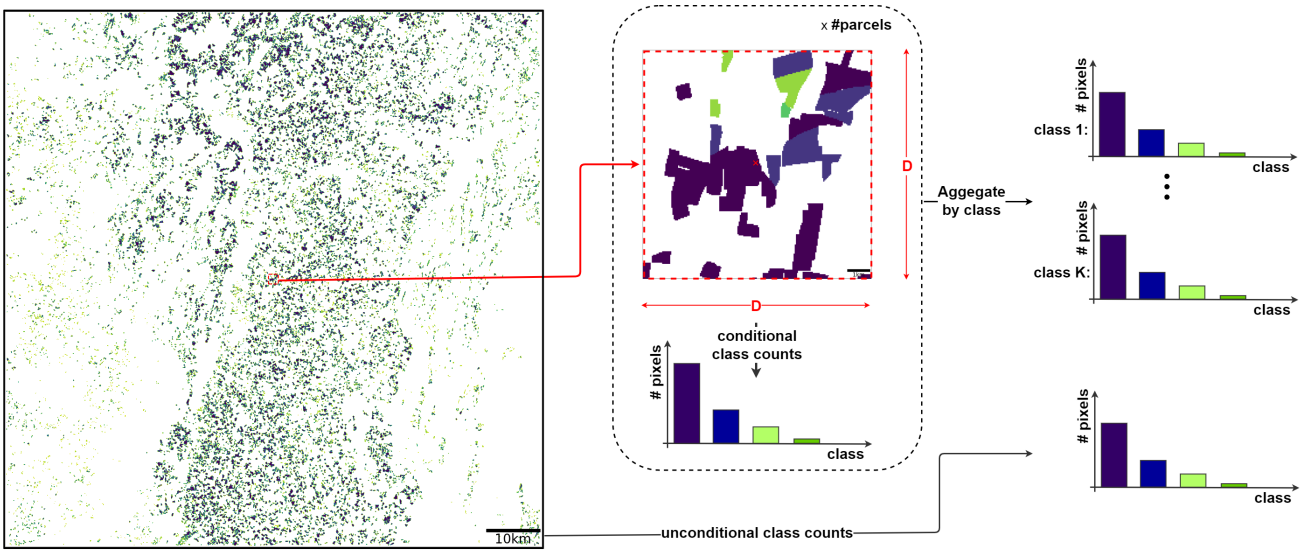

Figure 4. For each object id in our AOI, we find the distribution of crop types in a 1km square region around the center of th eobject. All such distributions are aggregated according to the crop type of the center object. We use the cosine similarity (Fig.5) between these distributions and unconditional pixel counts over the extent of the AOI as a measure of the spatial dependence between crop types.

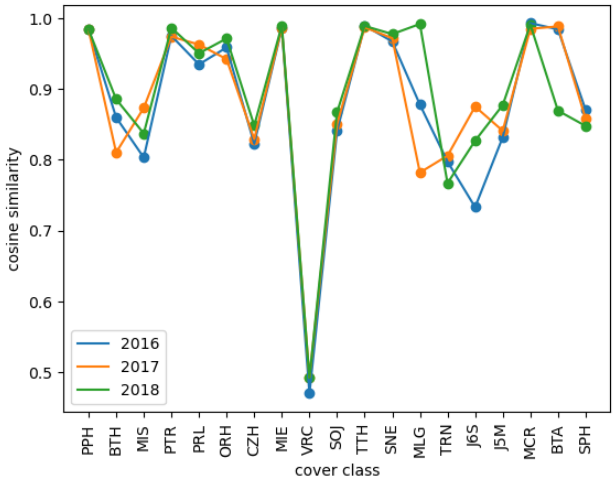

Figure 5. Cosine similarities between conditional and unconditional class counts.

area of interest we have obtained crop type ground truths in the form of geopolygons provided by the RPG[1] dataset. We we only keep crop types with more than 100k pixels and rasterize these polygons using [7] to create two maps for instance identities and crop types. For each object id we count the number of pixels per crop type in a 1km square region from the center of the object. All class pixel counts are then aggregated by crop type to get a list of 19 conditional crop type distributions. This process is presented schematically in Fig.4. We also calculate an unconditional distribution of pixels by counting the number of pixels per class over the extent of the AOI. We use the cosine similarity between class-conditional pixel counts and unconditional pixel counts as a meassure of the similarity between the these distribution. Results, visualized in Fig.5, show the cosine similarities to be close to 1 indicating strong correlation between the class-conditional and class-independent distribution of counts, which suggests that proximity to a specific class does not have a major effect on the distribution of crop types. We note how class VRC (vines) deviates from the remaining classes in that respect. However, a cosine similarity around 0.5 still indicates a high degree of correlation between the two distributions and, observing the confusion matrices in Fig.9 we note that VRC is actually one of the best predicted classes.

2. Training and evaluation

2.1. Training details

As discussed in section 4, for all experiments presented we train for the same number of epochs using the provided data splits from respective publications. The TSViT training schedule employs the AdamW optimizer [2] with a linear warmup of the learning rate from zero to a maximum value 10^{-3} at epoch 10, followed by cosine learning rate decay [4] down to $5 * 10^{-6}$ at the end of training. To exclude the effect of our training settings on performance metrics and avoid issues of convergence for other methods we also train using the suggested training settings from each respective

[1] https://www.data.gouv.fr/en/datasets/registre-parcellaire-graphique-rpg-contours-des-parcelles-et-ilots-culturaux-et-leur-groupe-de-cultures-majoritaire/

Ablation	settings		mIoU	#params (M)	IT (ms)
Factorization order	Spatial & Temporal		48.8	2.05	6.13
	Temporal & Spatial		**78.5**	1.66	4.67
#*cls* tokens	1		78.5	1.66	4.67
	K		**83.6**	1.66	5.78
Position encodings	Static		80.8	1.62	5.76
	Transformer encodings		81.2	2.44	12.45
	Date lookup		**83.6**	1.66	5.78
Interactions between *cls* tokens	temporal	spatial			
	✓	✓	81.5	1.66	9.82
	✓	✗	**83.6**	1.66	5.78
Patch size	2×2		84.8	1.66	11.8
	$\mathbf{3 \times 3}$		**83.6**	1.66	5.78
	4×4		81.5	1.66	3.85
	6×6		79.6	1.70	3.80
Feature dimension	64		81.2	0.57	4.41
	128		**83.6**	1.66	5.78
	256		83.7	5.41	9.45
MSA feature dimension	64		82.8	1.39	5.60
	128		**83.6**	1.66	5.78
	256		83.6	2.18	6.93
# MSA heads	1		82.5	1.66	5.78
	2		82.9	1.66	5.78
	4		**83.6**	1.66	5.78
	8		83.6	1.66	5.78
Training schedule	Adam-fixed		81.0	1.66	5.78
	Adam-exp.		82.5	1.66	5.78
	wAdam-cos(1warm)		82.8	1.66	5.78
	wAdam-cos(10warm)		**83.6**	1.66	5.78
Depth (100 epochs)	temporal	spatial			
	2	0	41.6	0.46	2.50
	4	0	42.7	0.86	4.30
	6	0	43.1	1.25	6.76
	8	0	42.8	1.65	8.59
	2	2	81.8	0.87	3.08
	4	2	83.3	1.26	5.43
	6	2	83.5	1.66	7.47
	8	2	83.6	2.05	9.42
	2	4	82.9	1.26	3.74
	4	4	83.4	1.66	5.78
	6	4	83.6	2.05	8.00
	8	4	83.9	2.45	10.02
	2	6	83.3	1.66	5.43
	4	6	83.5	2.05	6.94
	6	6	84.1	2.45	8.68
	8	**6**	**84.4**	2.84	10.70

Table 1. **Additional ablation on design choices for TSViT**. All models are trained using the Germany dataset for 150 epochs unless otherwise indicated (depth ablation). For each model we note its number of parameters (#params. $\times 10^6$) and inference time (IT) for a single sample with T=52, H,W=24 and C=13 size input on a Nvidia Titan Xp GPU.

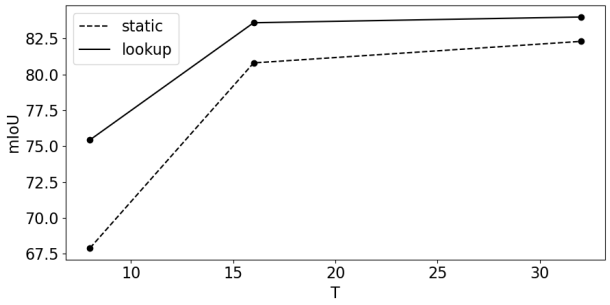

Figure 6. **Date lookup vs Static position encodings** for a varying number of timestamps T in Germany.

study and report best performances among training rounds for each model. In particular:

- for Germany we train for 150 epochs with a learning rate 10^{-4} and exponential weight decay with a factor 0.975 applied every two epochs according to [6].
- for T31TFM16-18 we train for 100 epochs with a learning rate 10^{-4} and exponential weight decay with a factor 0.975 applied every two epochs according to [6].
- for PASTIS we train for 100 epochs with a constant learning rate 10^{-3} according to [1].

We always use the largest number of samples that can fit into $\times 2$ Nvidia Titan Xp GPUs in a data parallel fashion. Using a patch size $h = w = 2$ and four layers for the temporal and spatial encoders this is 32 for Germany, 16 for PASTIS (includes more acquisition times) and 6 for T31TFM-1618 (larger sample size 48). For the T31TFM-1618 dataset a model with 8 temporal and 4 spatial encoder layers, patch size $h = w = 3$, and same settings otherwise, can be trained with batch size 16 and reach similar performances (mIoU 62.8% vs 63.1%).

2.2. Additional ablations

In section 4.2 we presented the results of an ablation study on the most important design choices for TSViT using the Germany dataset [5]. These results are also presented here, in the top part of Table 1. Additional results are presented in the bottom part of Table 1. To obtain these numbers our design utilises all choices that were found to benefit performance in section 4.2, excluding the patch size, we now use (3×3) patches to increase the speed of experiments. In addition to ablation results, we also present the number of parameters and inference time for each model tested to show how our design choices affect these values. To calculate the inference time we assume a single sample input with size T=52, H,W=24, C=13 for a direct comparison with inference times presented in Table 2 in the main paper. Inference time is measured as the average of 300 repetitions following a warm up period for the GPU. During each repetition we synchronize the GPU with the CPU such that time is measured only after the process running on the GPU has been completed. We test the effect of our date-specific **position encodings** compared to a fixed set of values and find a significant -2.8% performance drop from using fixed size $\mathbf{P_T}$ compared to our proposed lookup encodings. Our position encoding module benefits more compared to a static one when there is a lot of variation in timestamps as each static encoding will need to represent a wider range of dates. If there is no variation in dates among samples then each static encoding will only need to represent a single date making the two approaches equivalent. To show this effect we in/decrease the variation in sample dates by selecting different number of timestamps (T) for each training session. Decreasing T allows for more combinations of dates, leading to increased variation in terms of timestamps seen per position encoding. As shown in Fig.6 improvements are indeed more pronounced for small values of T. We further test the capacity of a small Transformer model in generating dynamic temporal position encodings from one-hot day-of-year (doy) encodings. This increases the model's parameter count and inference time but is ultimately outperformed by the lookup encodings. Using (3×3) patches, in order to retain an uncompressed representation of the size $3 \times 3 \times 13 = 117$ inputs, we will need to employ a **feature dimension** greater or equal to that value. We find that $d = 64$ leads to a noticeable performance drop, while $d = 256$ brings no clear benefits, so we retain $d = 128$ going forward. Regarding the **MSA feature dimension**, again, we find that an inverted bottleneck design with 256 features brings no performance gains w.r.t. size 128 MSA features. A bottleneck design with half the number of features (64) underperforms both options. We proceed with 128 features for the MSA operations. Training our model with a varying **number of MSA heads**, we find that while using too few heads is suboptimal, there are little gains from increasing that number beyond four, thus, we proceed with our initial design. Regarding the **choice of optimization algorithm**, apart from our proposed settings, commonly used for training Transformers, we also test the effect of training settings typically used in the land cover and crop recognition literature. These consist of using the Adam [3] optimizer and a fixed learning rate [1] or exponential decay [6]. We find that both settings reach suboptimal performance compared to our proposed settings. Additionally, we find that this can mostly be attributed to the linear learning rate warmup employed in the begginning of training. More specifically, we start with a zero value for the learning rate and linearly increase it up to a predefined value at a specific time during training; after this point the learning rate decreases following a one cycle cosine decay. We

observe that reducing the number of warmup epochs from 10 to 1 causes a significant reduction in performance by -0.8% mIoU which is very close to what is achieved by an exponential decay setting with no warmup. Finally, we do a full factorial design on **encoder depths** using $[2, 4, 6, 8]$ and $[0, 2, 4, 6]$ layers for the temporal and spatial encoders. We observe that not using a spatial encoder ($L = 0$) leads to very large performance drops independent of the temporal encoder depth. Excluding these results, we find that the best performance predictor is the total number of layers and that the depth of each submodule has a similar effect.

2.3. Additional results

In Figs.7, 8 and 9 we respectively show confusion matrices for Germany, PASTIS and the T31TFM-1618 datasets. In Germany, all classes are predicted with a high degree of accuracy. We observe some degree of confusion between crop types that grow during the same period, e.g. "winter rye", "winter wheat", "winter spelt", "winter barley" or "summer oat", "summer barley". In PASTIS and T31TFM-1618, model performance is significantly lower, however, the main driver of performance degradation with respect to Germany can be attributed to some bad performing classes, especially so for the T31TFM-1618 dataset. For a qualitative assessment of semantic segmentation models, we present illustrations of semantic segmentation predictions for the three best models in Germany, PASTIS and T31TFM-1618 in Figs.10, 11 and 12.

References

[1] Vivien Sainte Fare Garnot and Loic Landrieu. Panoptic segmentation of satellite image time series with convolutional temporal attention networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4872–4881, October 2021. 1, 2, 5

[2] Loshchilov Ilya, Hutter Frank, et al. Decoupled weight decay regularization. *Proceedings of ICLR*, 2019. 3

[3] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014. 5

[4] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. 3

[5] Marc Rußwurm and Marco Körner. Multi-temporal land cover classification with sequential recurrent encoders. *ISPRS International Journal of Geo-Information*, 7(4):129, Mar 2018. 1, 2, 5

[6] Michail Tarasiou, Riza Alp Güler, and Stefanos Zafeiriou. Context-self contrastive pre-training for crop type semantic segmentation. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–1, 2022. 1, 2, 5

[7] Michail Tarasiou and Stefanos Zafeiriou. Deepsatdata: Building large scale datasets of satellite images for training machine learning models. In *IGARSS 2022 - 2022 IEEE International Geoscience and Remote Sensing Symposium*, pages 4070–4073, 2022. 3

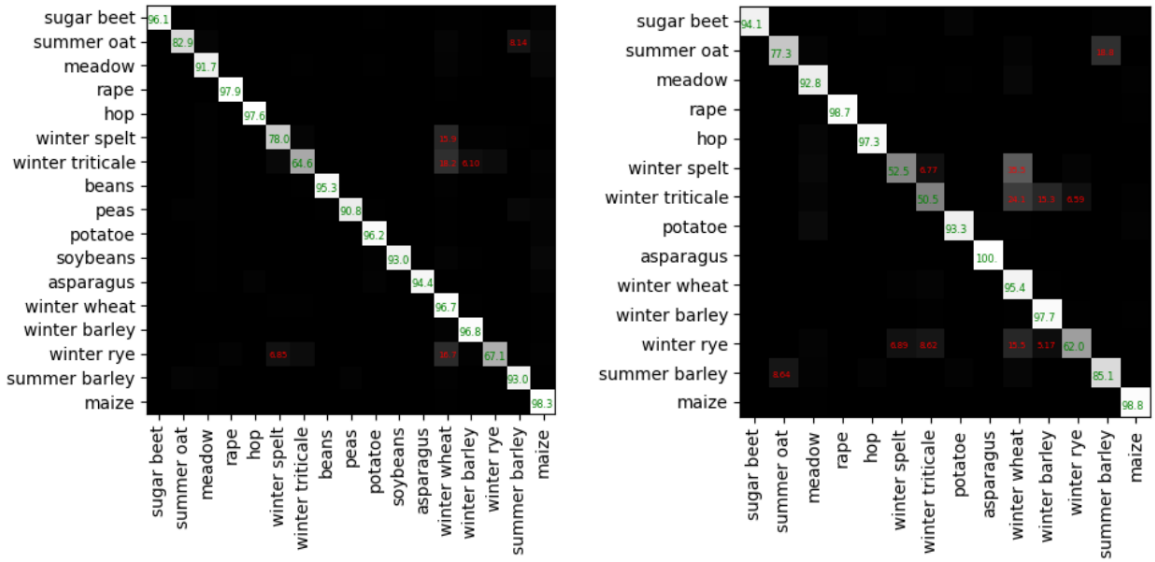

Figure 7. **Confusion matrices for Germany. (left)** Semantic segmentation. **(right)** Object classification. We explicitly show numerical values for diagonal elements (green) and non-diagonal elements with error $\geq 5\%$ (red).

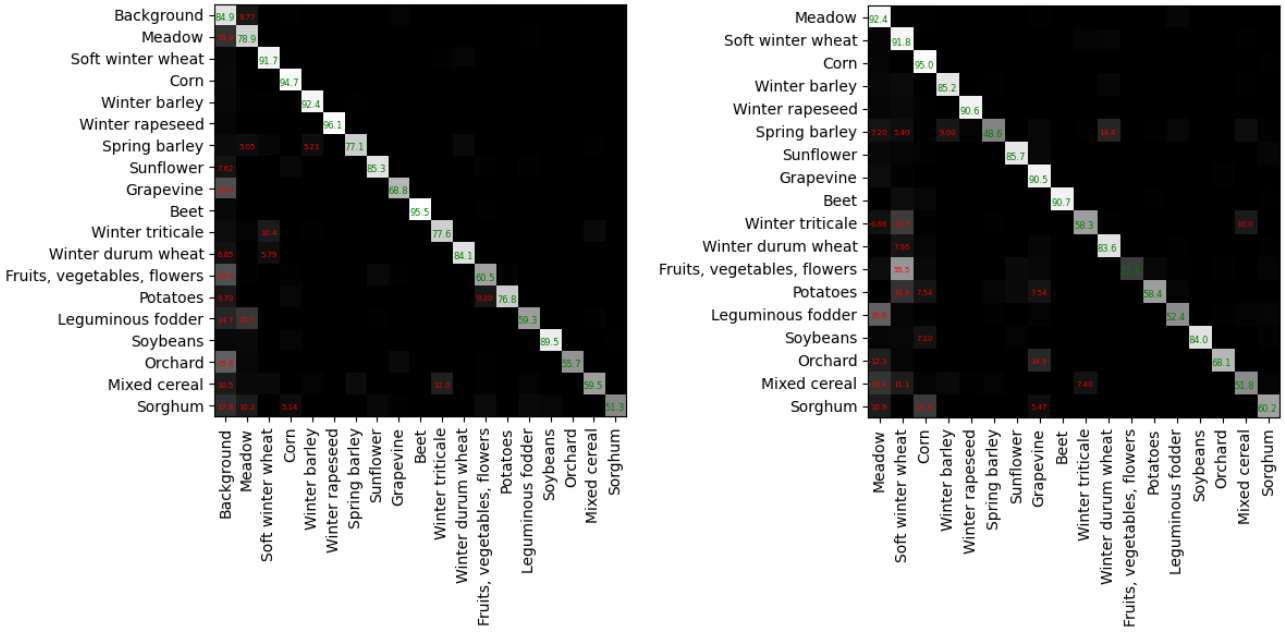

Figure 8. **Confusion matrices for PASTIS. (left)** Semantic segmentation. **(right)** Object classification. We explicitly show numerical values for diagonal elements (green) and non-diagonal elements with error $\geq 5\%$ (red).

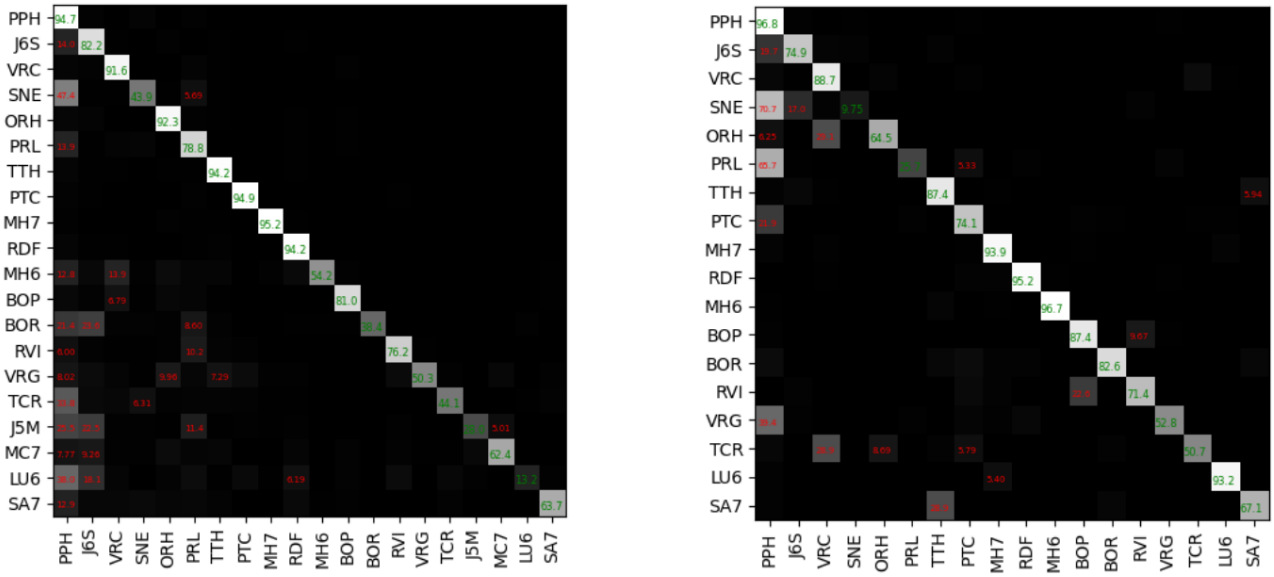

Figure 9. **Confusion matrices for T31TFM-1618. (left)** Semantic segmentation. **(right)** Object classification. We explicitly show numerical values for diagonal elements (green) and non-diagonal elements with error $\geq 5\%$ (red).

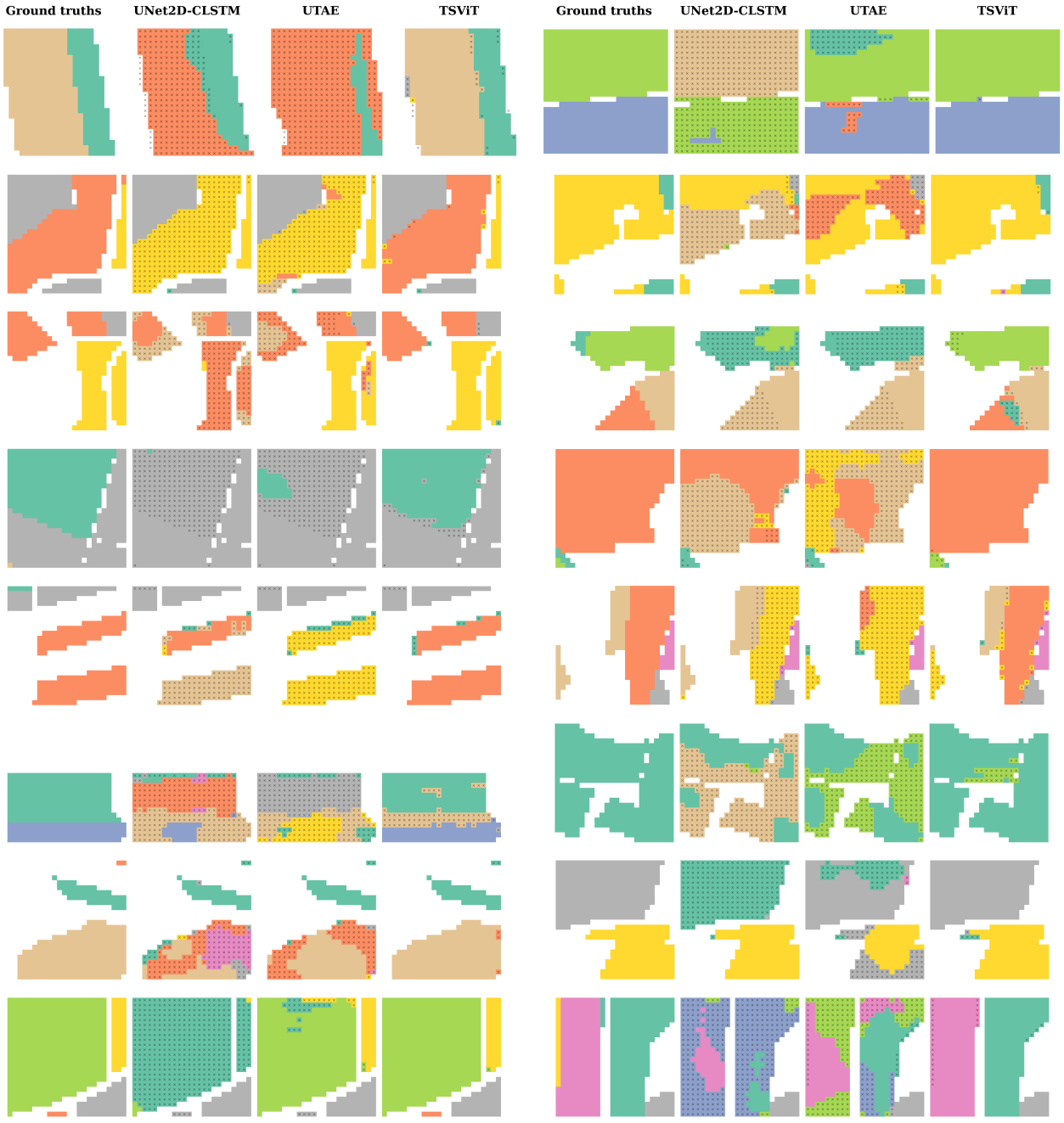

Figure 10. **Qualitative examples for Germany.** Black ”x” indicates a false prediction at that particular location.

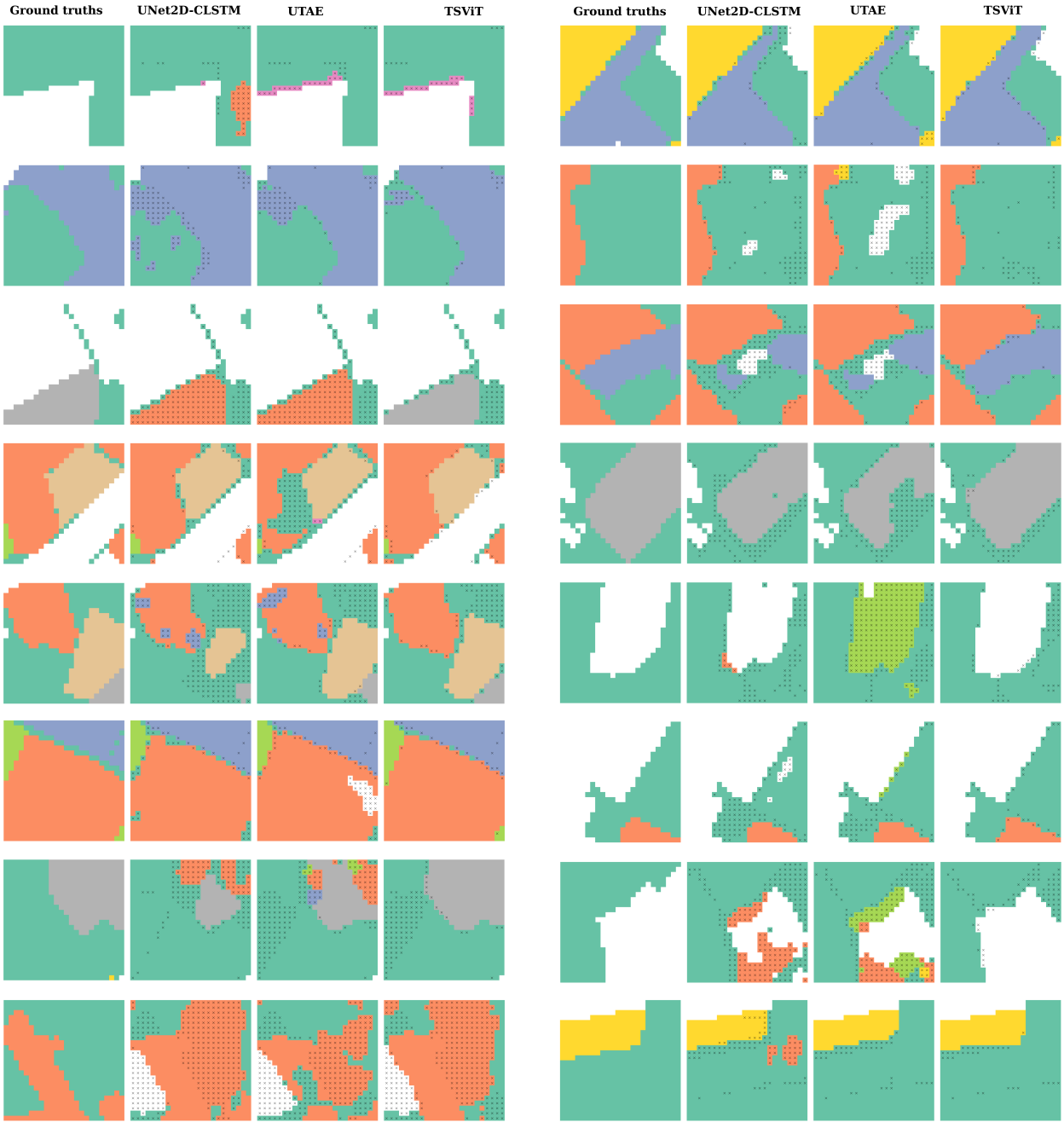

Figure 11. **Qualitative examples for PASTIS.** Black ”x” indicates a false prediction at that particular location.

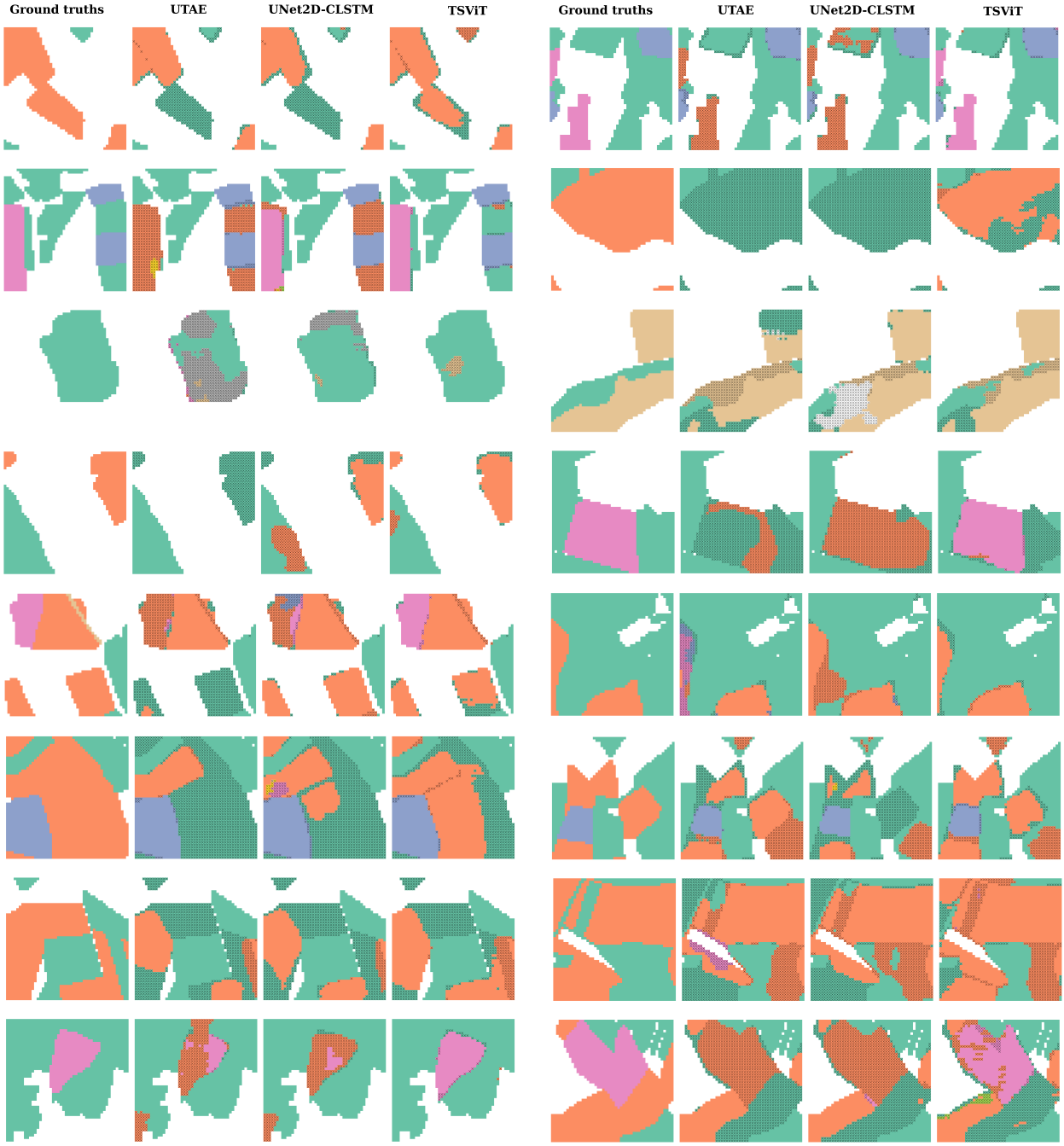

Figure 12. **Qualitative examples for T31TFM1618.** Black ”x” indicates a false prediction at that particular location.