EPFL

Image processing for Earth Observation

3e Sequence modeling

Devis TUIA

EPFL, fall semester 2025

École polytechnique fédérale de Lausanne

Content (6 weeks)... almost there!

- W1 General concepts of image classification / segmentation
  Traditional supervised classification methods (RF)
- W2 Traditional supervised classification methods (SVM)
  Best practices

- W3 Elements of neural networks
- W4 Convolutional neural networks
- W5 Convolutional neural networks for semantic segmentation
- W6 Sequence modeling, change detection

What about the temporal dimension?

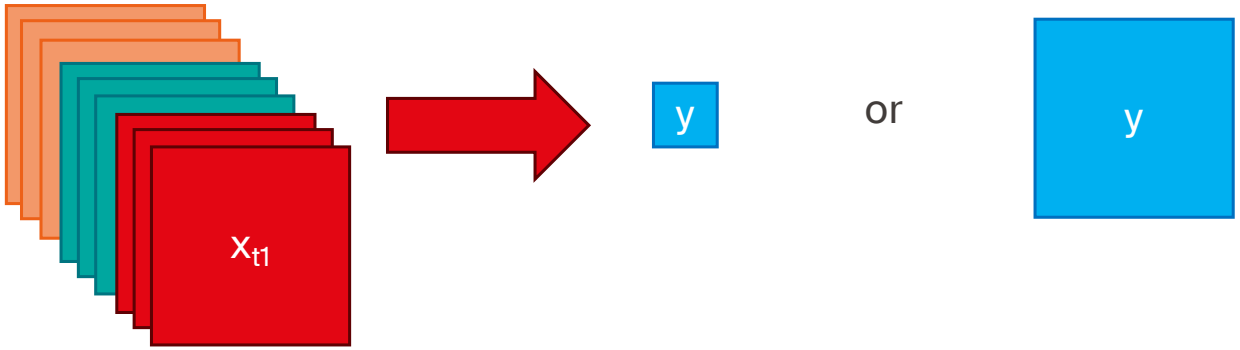
What about the temporal dimension?

- We haven't been talking much about it.
- We focused mostly on spatial context and single images classification and segmentation

- Today we talk about what can be done for time series (or image sequences)

Time can have several uses in RS classification

1 - as extra features to classify or segment

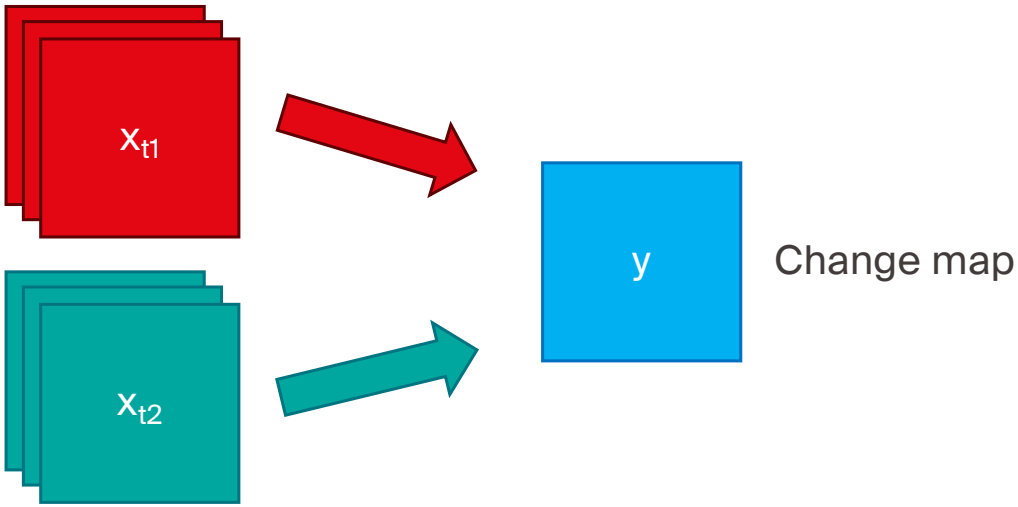


- How to exploit the temporal sequence?
- How to fuse the different information?

D. Tuia. ECEO

Time can have several uses in RS classification

2 - as a sequence to monitor changing processes: between 2 dates

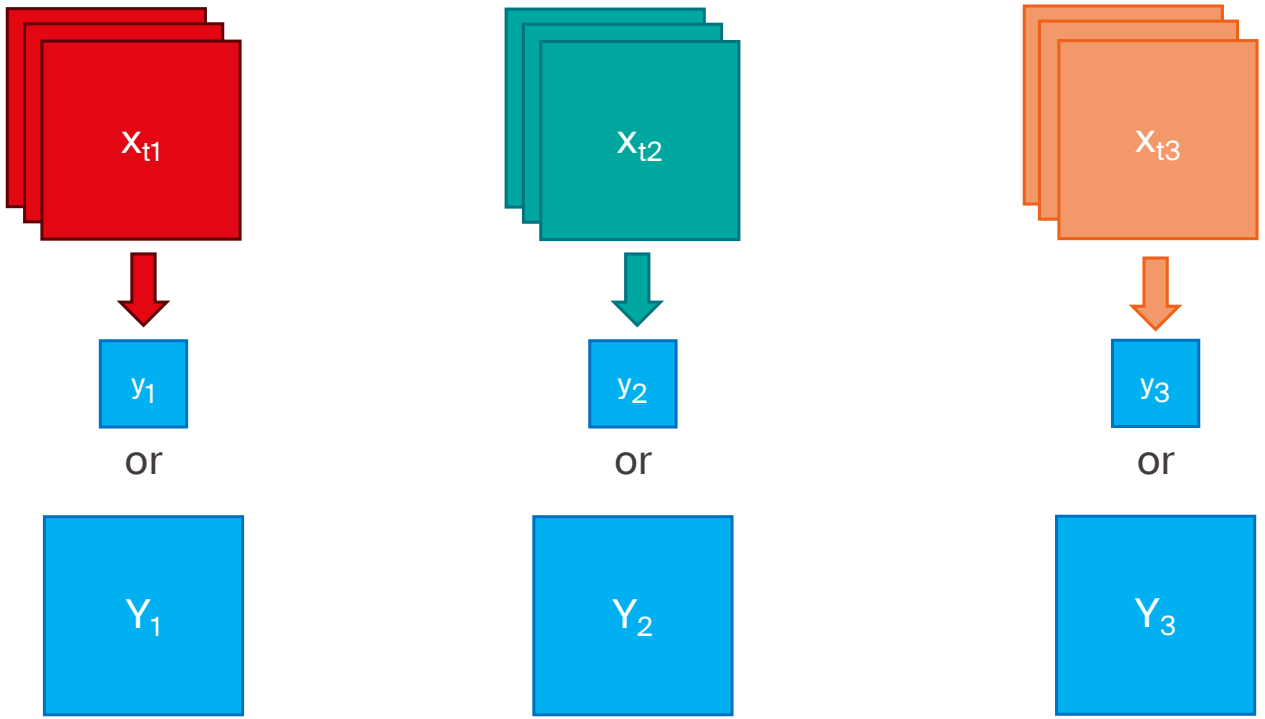


- Should the two models be the same?
- We have very little label information...

D. Tuia. ECEO

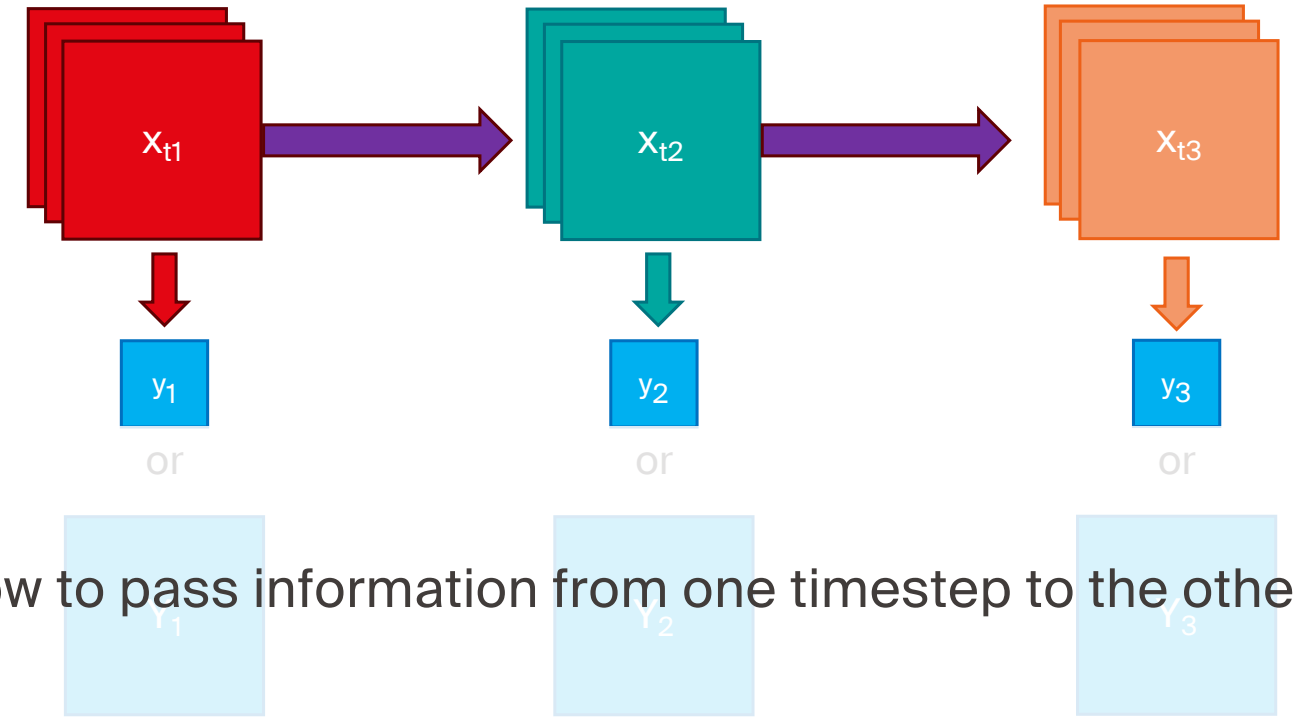
Time can have several uses in RS classification

- 3 - as a sequence to monitor (changing) processes

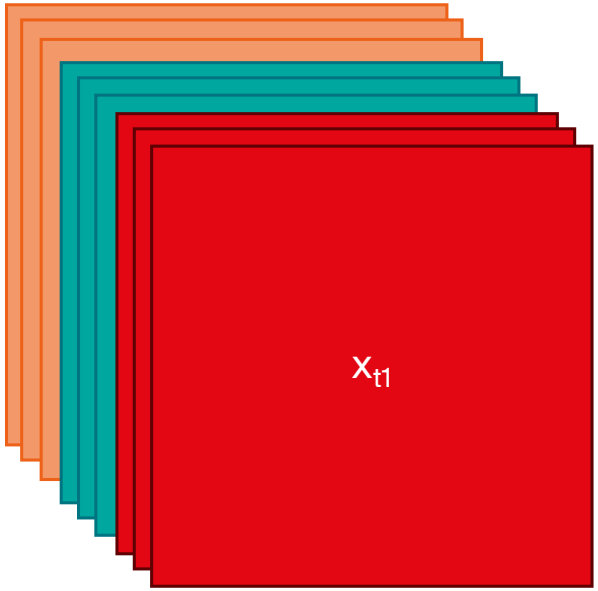


Time can have several uses in RS classification

- 3 - as a sequence to monitor (changing) processes: monitoring



- How to pass information from one timestep to the other?

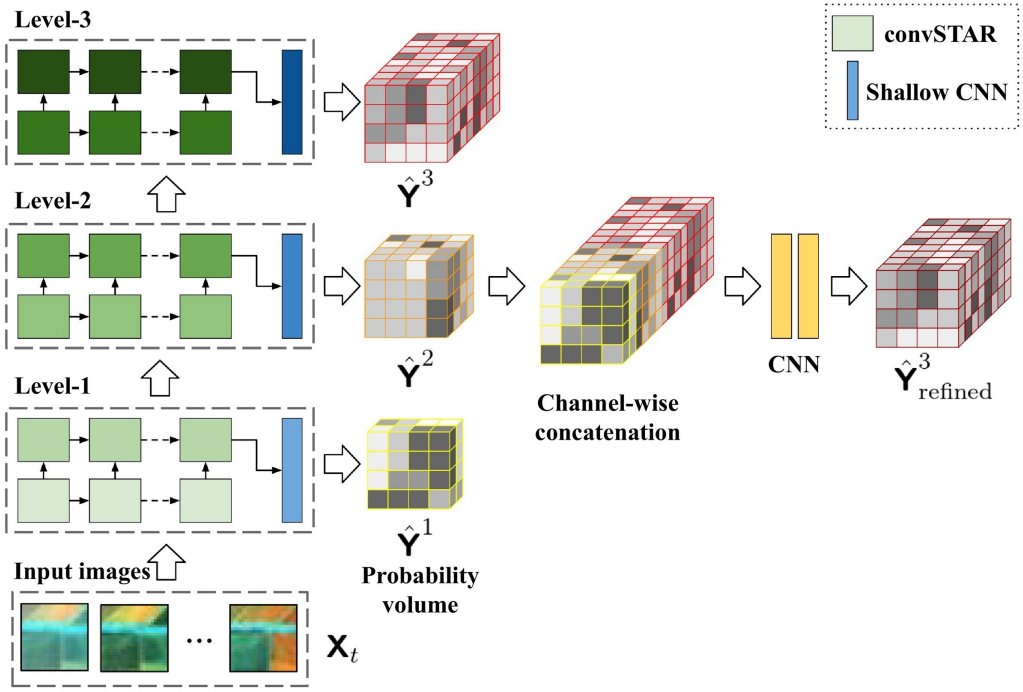


Classify a static process with multiple time steps

It’s all about fusion

Stacking time information

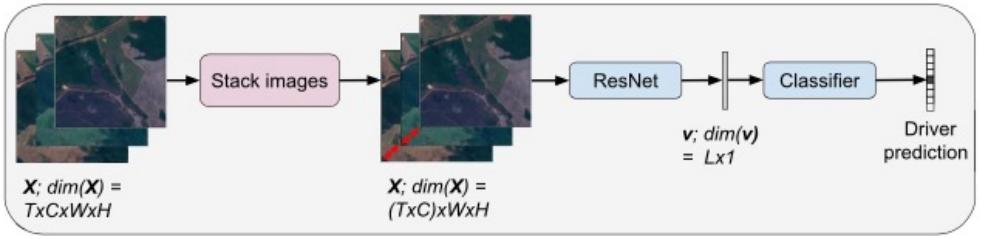
- Using multiple timesteps brings more information to the classifier
- Quite used in crops classification



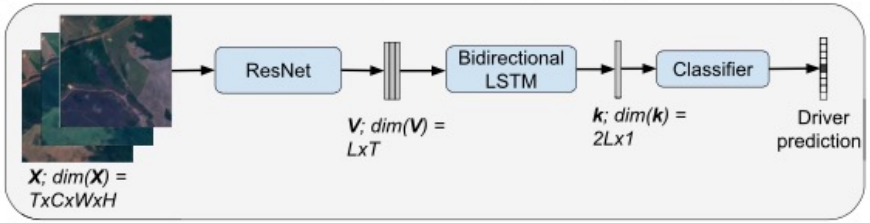
From Turkoglu et al., Remote Sensing of Environment, 2021
https://www.sciencedirect.com/science/article/pii/S0034425721003230

Passing the information

- A more advanced procedure would be to take the information of a timestep and pass it on to the next one
- We call this a sequence model (here a LSTM is used, more about that later in this course.)
- Here we classify drivers of deforestation in the tropics with a “stacked” approach (baseline 1) vs a LSTM (baseline 2).

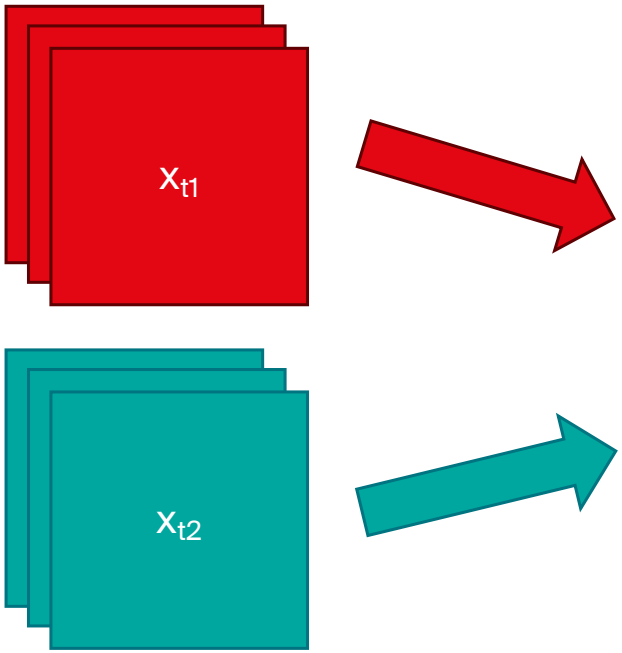


(a) Baseline (1): CNN. If the input is a single image, no stacking is done



(b) Baseline (2): CNN-LSTM

J. Pisl, M. Russwurm, L. Hughes, G. Lenczner, L. See, J. D. Wegner, and D Tuia. Mapping drivers of tropical forest loss with satellite image time series and machine learning. *Enviro. Res. Lett.*, 19(6):064053, 2024. https://iopscience.iop.org/article/10.1088/1748-9326/ad44b2



Change detection

Classify a changing process with two time steps

Siamese networks and representation learning

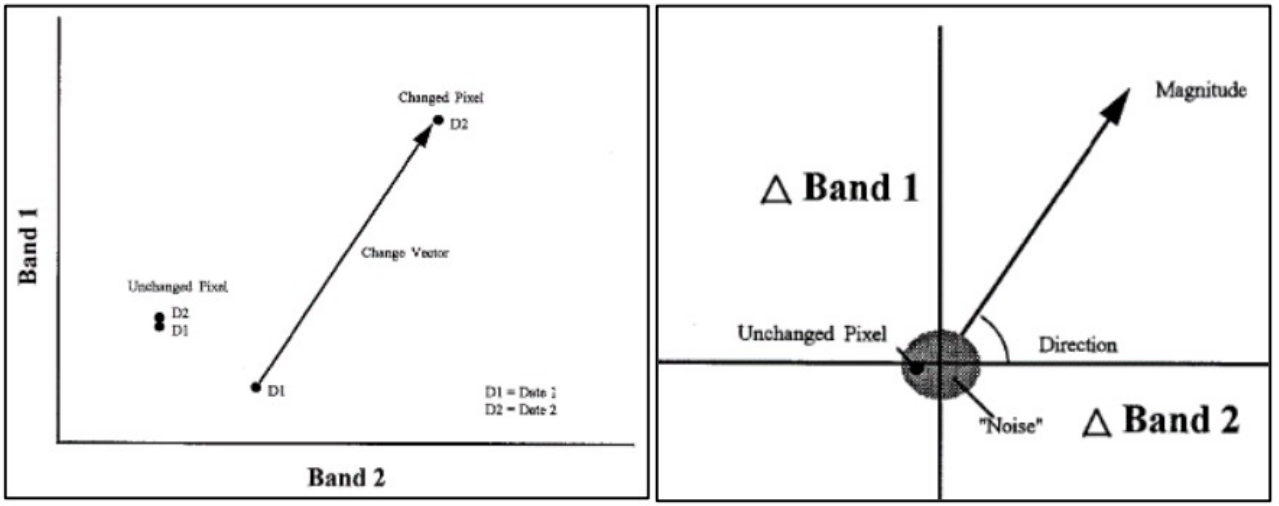
There are seven total changes. Gergely Dudás/Dudolf

Change detection as the 7 differences game

- When dealing with remote sensing, we face a similar problem
  - Changes are rare
  - Changes are small
  - Most of the image hasn't changed

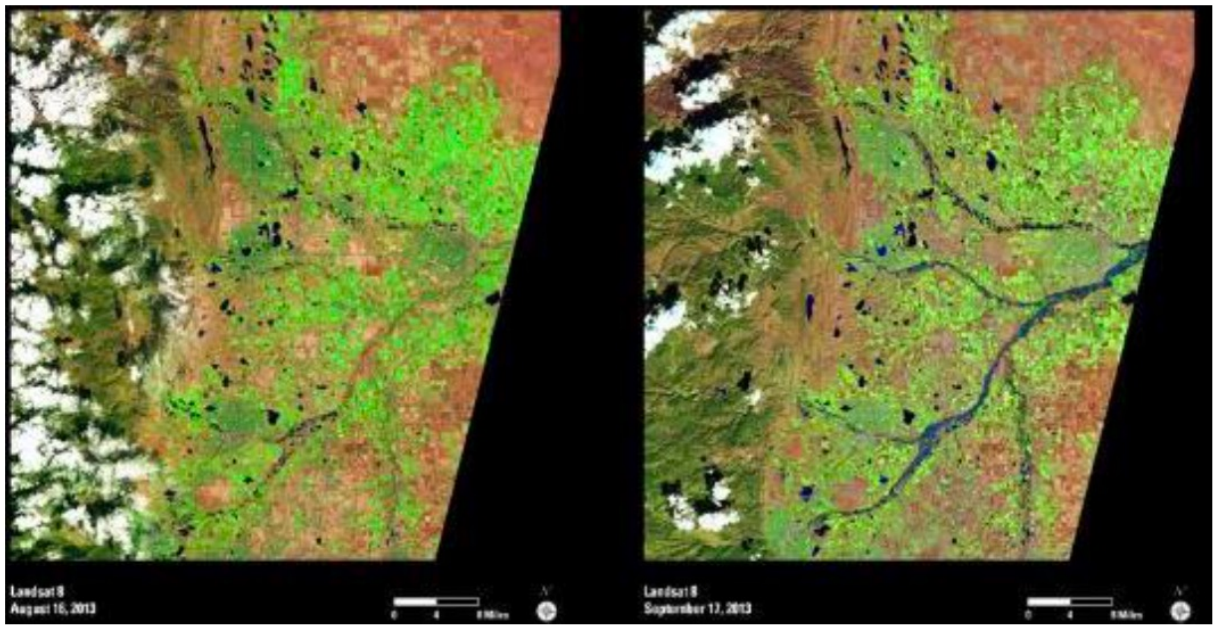
First methods were unsupervised

- Change vector analysis (Bovolo et al. , 2005)



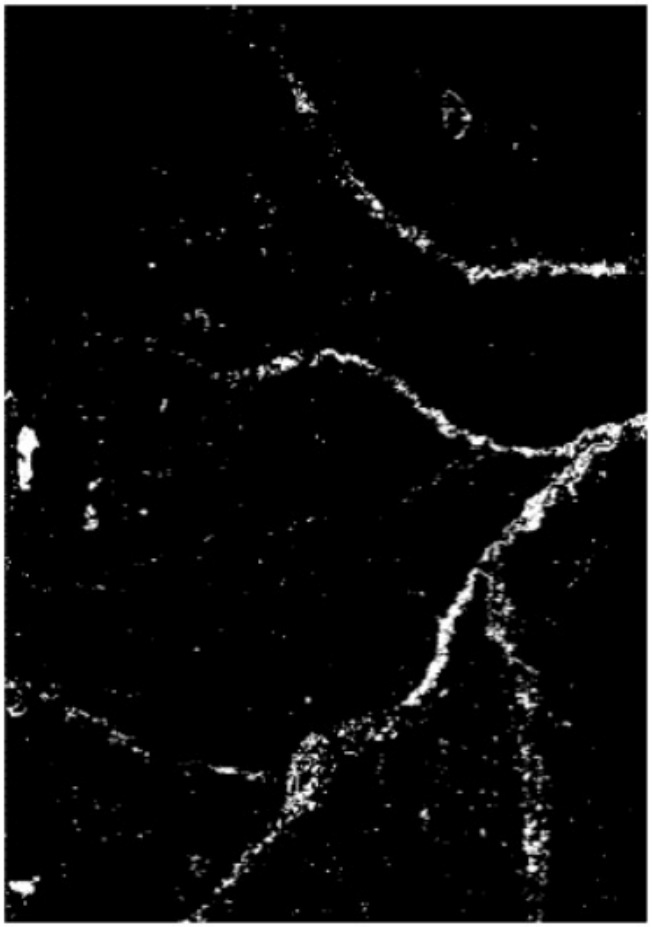
CVA in action

- Example: colorado flood in August 2013

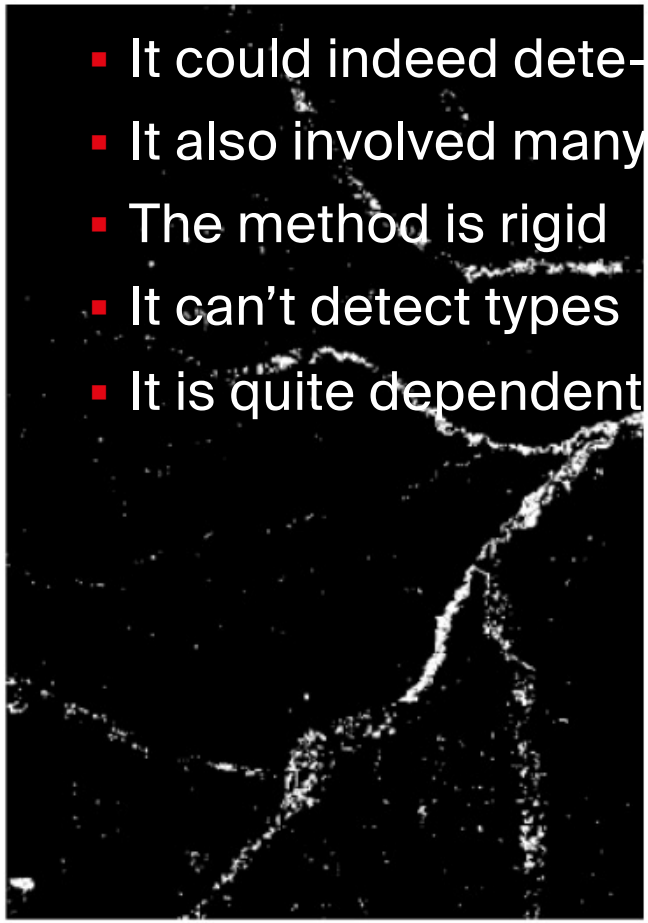


CVA in action

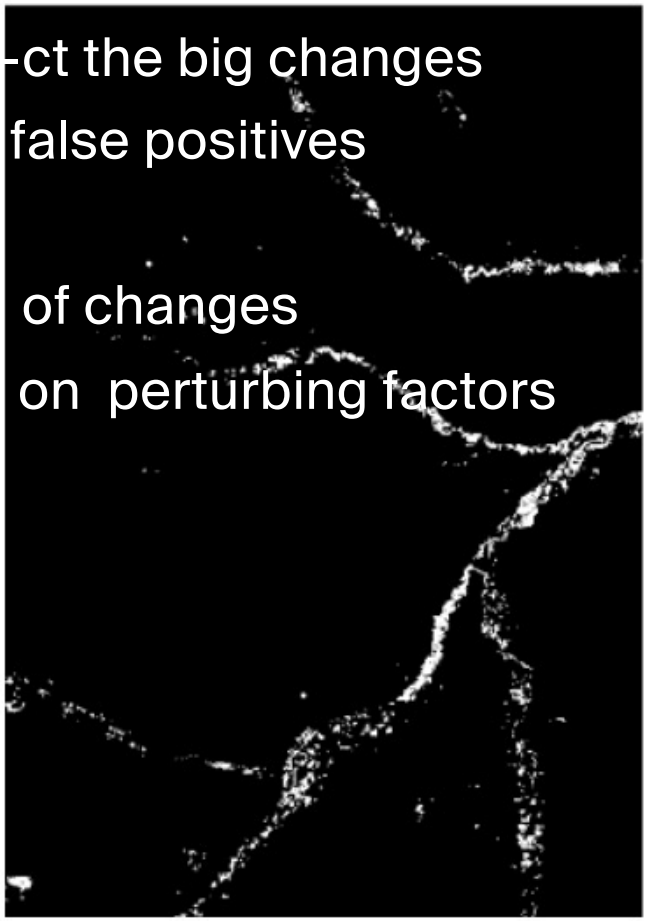
- It could indeed dete-–ct the big changes
- It also involved many false positives
- The method is rigid
- It can’t detect types of changes
- It is quite dependent on perturbing factors



CVA



CVA after postprocessing



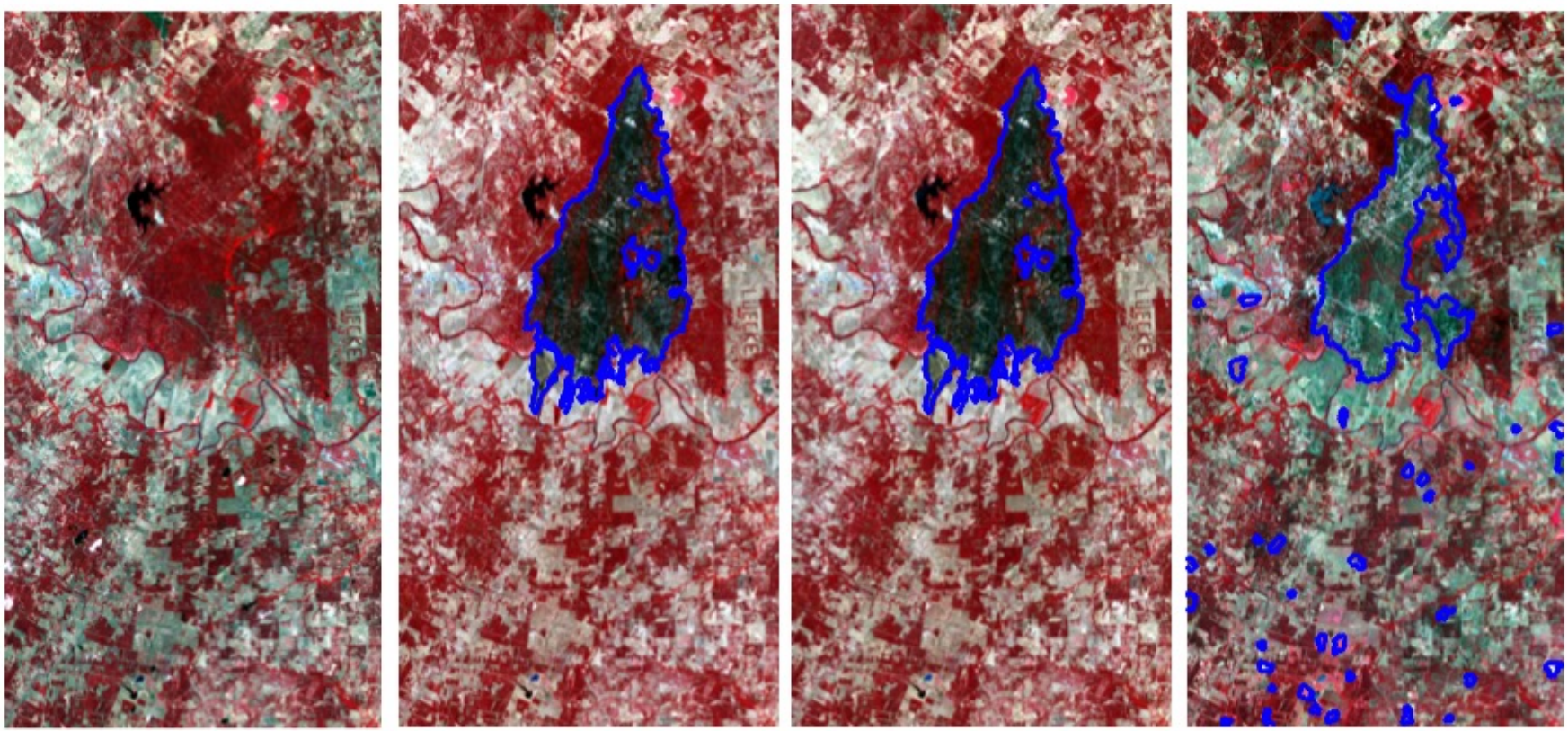
CVA after more postprocessing

Change detection NOT as the 7 differences game

- When dealing with remote sensing, we face a similar problem
  - Changes are rare
  - Changes are small
  - Most of the image hasn't changed

- Many things that have not changed, actually look very different
  1. Seasonal effects: grass growing

Seasonal changes



(a) (b) (c) (d)

26 August 2011 Landsat 5	11 September 2011 Landsat 5	12 September 2011 ALI	28 September 2011 Landsat 8

Change detection NOT as the 7 differences game

- When dealing with remote sensing, we face a similar problem
  - Changes are rare
  - Changes are small
  - Most of the image hasn't changed

- Many things that have not changed, actually look very different
  1. Seasonal effects: grass growing
  2. Illumination effects: a color can look very different from an image to the other

When illumination changes

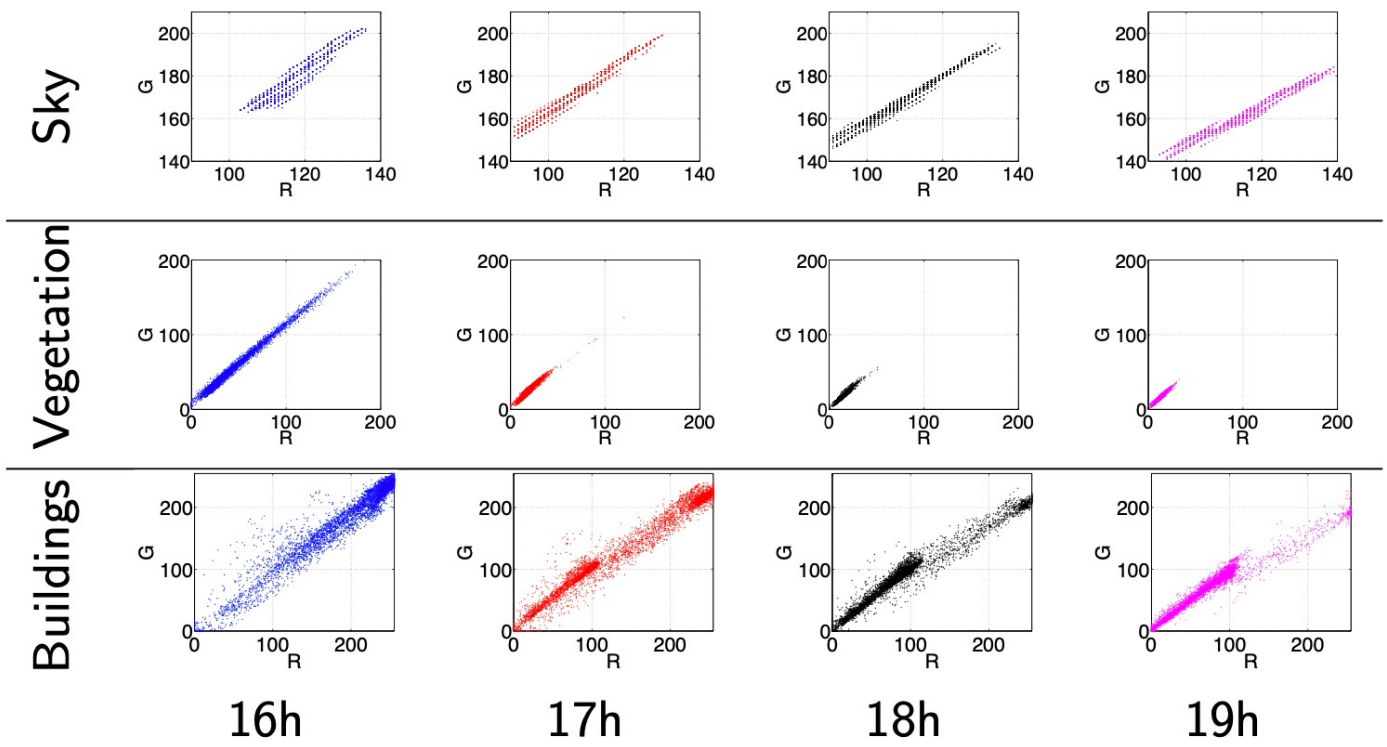
16h

17h

18h

19h

Effect on spectral signals



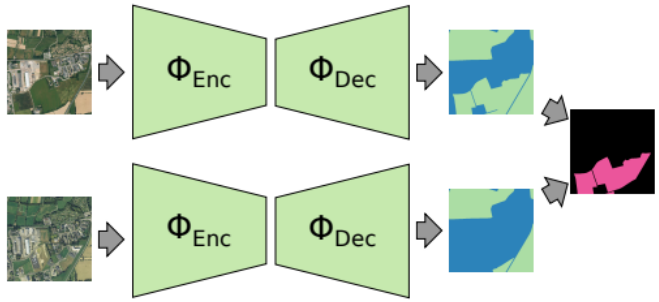
So what can we do?

1 Use a supervised method

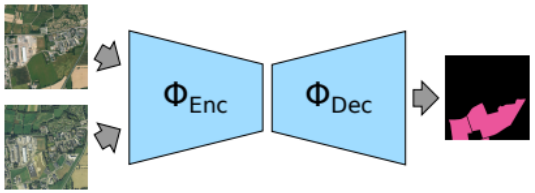
2 Make the image spaces more similar before applying an unsupervised method

1. Use Supervised methods

Supervised approaches to change detection



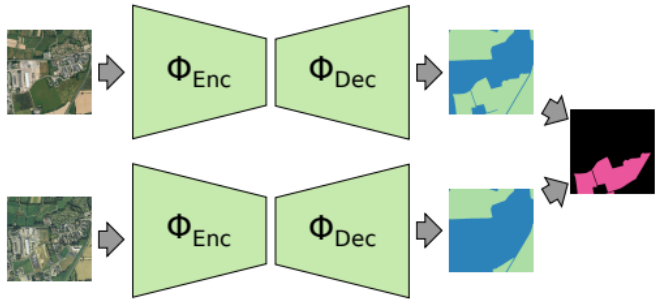
(a) Strategy 1: semantic CD from land cover maps.



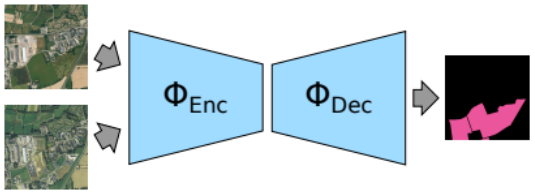
(b) Strategy 2: direct semantic CD.

[From Daudt Et al., 2018]

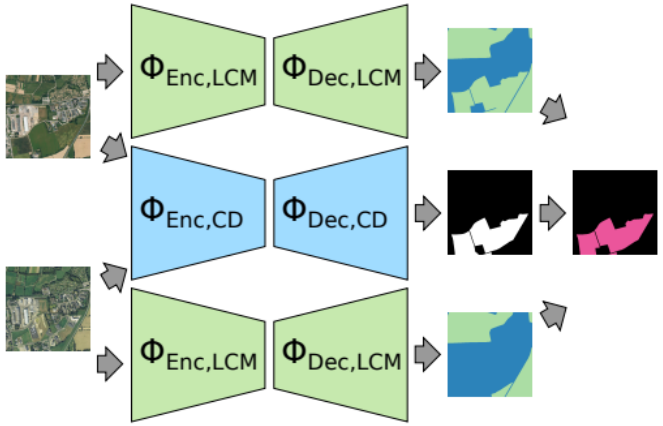
Supervised approaches to change detection



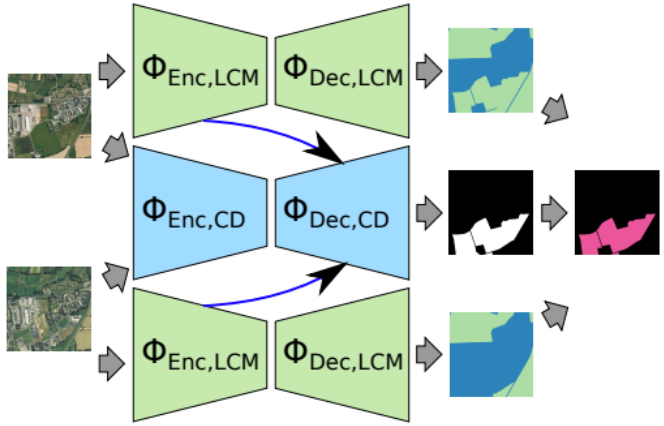
(a) Strategy 1: semantic CD from land cover maps.



(b) Strategy 2: direct semantic CD.



(c) Strategy 3: separate CD and LCM.



(d) Strategy 4: integrated CD and LCM.

[From Daudt Et al., 2018]

Supervised approaches to change detection

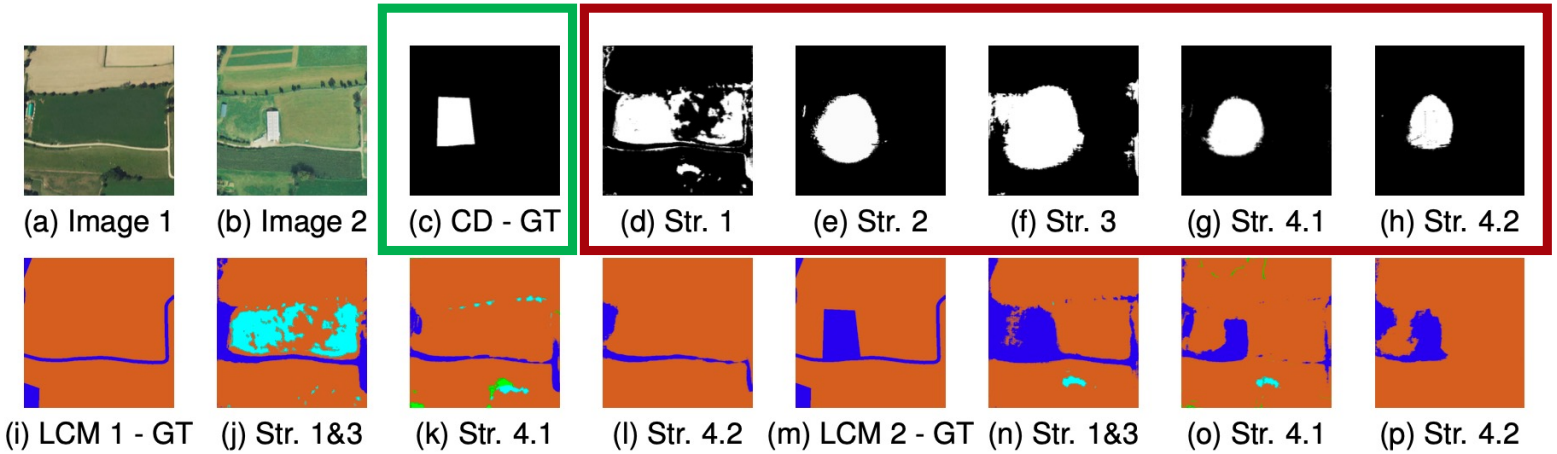


Figure 4.6: Illustrative images of the obtained results: (a)-(b) multitemporal image pair; (c) ground truth change detection map; (d)-(h) predicted change maps; (i)-(l) ground truth and predicted land cover maps for image 1; (m)-(p) ground truth and predicted land cover maps for image 2.

	CD			LCM	
	Kappa	Dice	Tot. acc.	Kappa	Tot. acc.
Strategy 1	3.99	5.56	86.07	**71.92**	87.22
Strategy 2	21.54	-	**98.30**	-	-
Strategy 3	12.48	13.79	94.72	**71.92**	87.22
Strategy 4.1	19.13	20.23	96.87	67.25	85.74
Strategy 4.2	**25.49**	**26.33**	98.19	71.81	**89.01**
CNNF-O	0.74	2.43	64.54	-	-
CNNF-F	3.28	4.84	88.66	-	-
PCA+KM	0.67	2.31	83.95	-	-

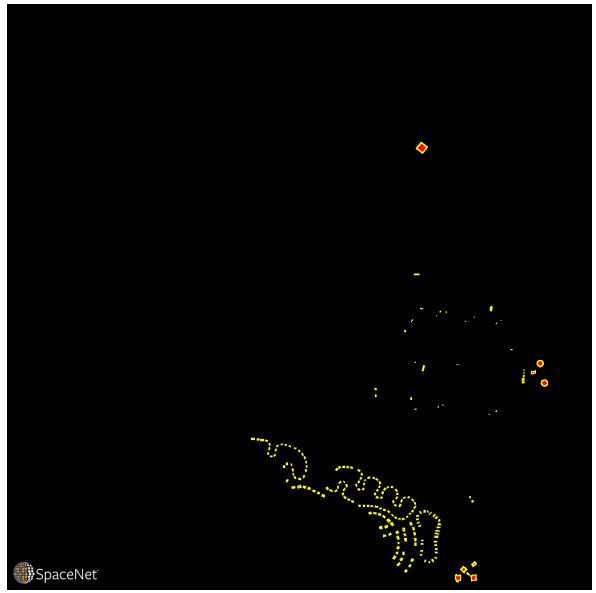
[From Daudt Et al., 2018]

What do we need to go further

- Probably more data,
  - with more diversity
  - with more samples
  - in geographies not too different from the ones under study

- The good news:

such data start to appear. See SpaceNet7 →



[Source: https://spacenet.ai/sn7-challenge/]

Still the problems are difficult!

Competitor	Final Place	Overall Score	Architectures	# Models	Pre-Training Weights	Training Time (H)	Inference Rate
lxastro0	1	41.00	1 × HRNet	1	ImageNet	36	346 km² / min
cannab	2	40.63	3 × EfficienNet-b6 + UNet (siamese) 3 × EfficientNet-b7 + UNet (siamese)	6	None	23	49 km² / min
selim_sef	3	39.75	1 × EfficienNet-b6 + UNet 3 × EfficientNet-b7 + UNet	4	None	46	87 km² / min
motokimura	4	39.11	10 × EfficienNet-b6 + UNet	10	ImageNet	31	42 km² / min
MaxsimovKA	5	30.74	1 × SENet154 +UNet (siamese)	1	ImageNet	15	40 km² / min
baseline	N/A	17.11	1 × VGG16 + UNet	1	None	10	375 km² / min

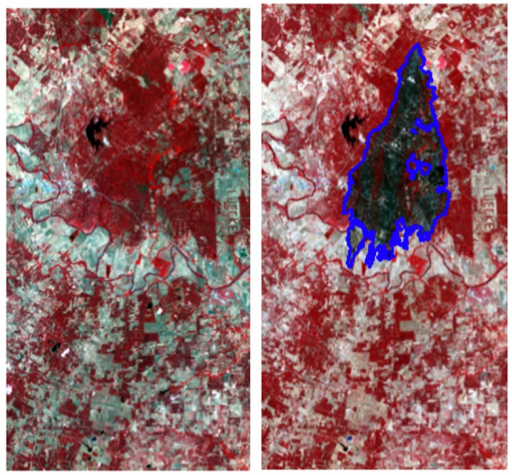
Table 1. SpaceNet 7 Winners

[Source: https://spacenet.ai/sn7-challenge/]

2. Make the image spaces more similar before applying an **unsupervised** method

The idea is simple

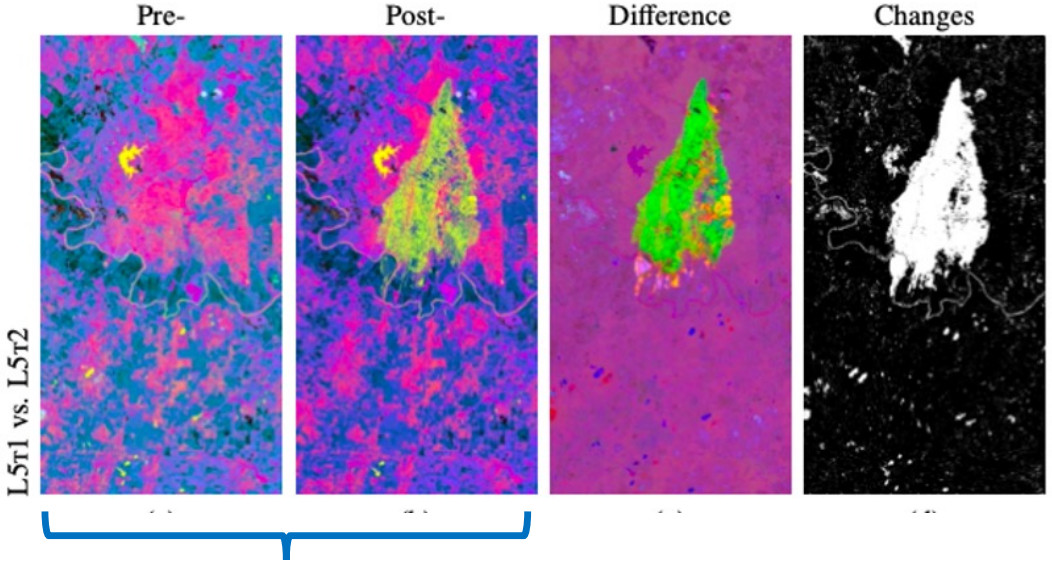
- Modify the images in a way that
  - all the unchanged parts look similar
  - only the changes are highlighted



(a) (b)

26 August 2011
Landsat 5

11 September 2011
Landsat 5



Same two images after projection

In input space, unchanged areas look very different

The idea is simple

- Modify the images in a way that
  - all the unchanged parts look similar
  - only the changes are highlighted

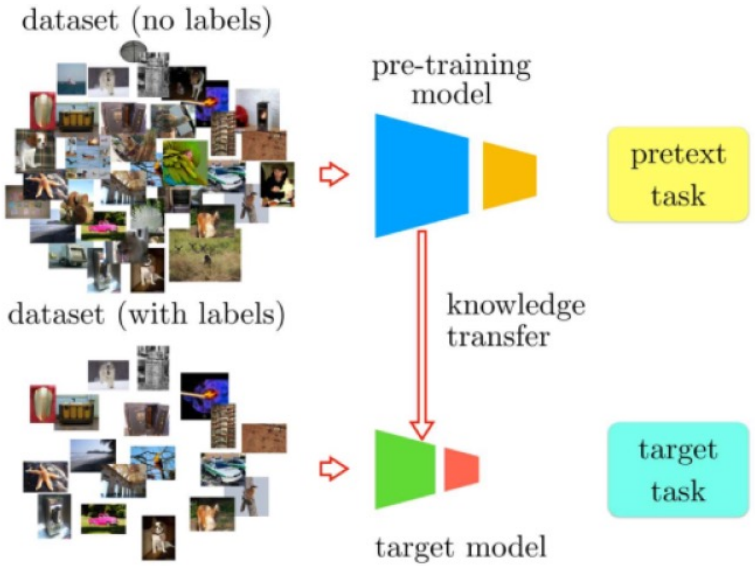
- We want to lear a representation that makes unchanged areas more similar.
- But we don't have labels at all!
- We learned with **self supervised learning,** a.k.a

  learning from a tasks that is not the one you want, but for which you can get the labels for free.

Leenstra, M. , Marcos, D., Bovolo, F., **Tuia, D.**, Self-supervised pretraining enhances change detection in Sentinel-2 images Pattern Recognition in Remote Sensing workshop, ICPR, 2021

What is Self Supervised learning?

- A new learning paradigm
- Learning model via a pretext task
  - A task for which the labels can be extracted automatically from the data
  - A task that is connected to the main task. So learning it helps the main task

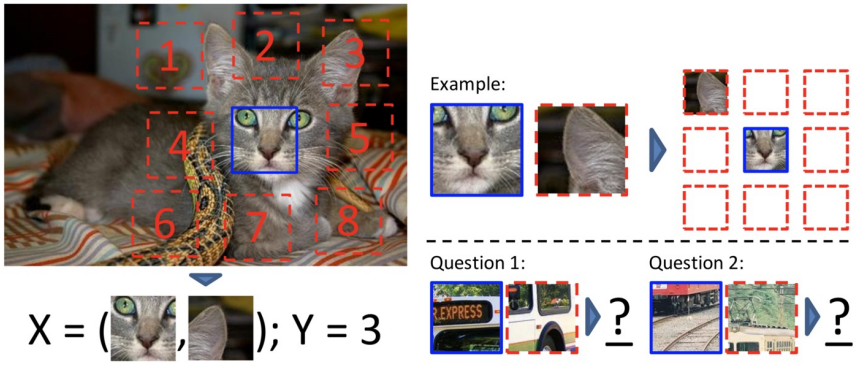


Groundai.com

What is a pretext task?

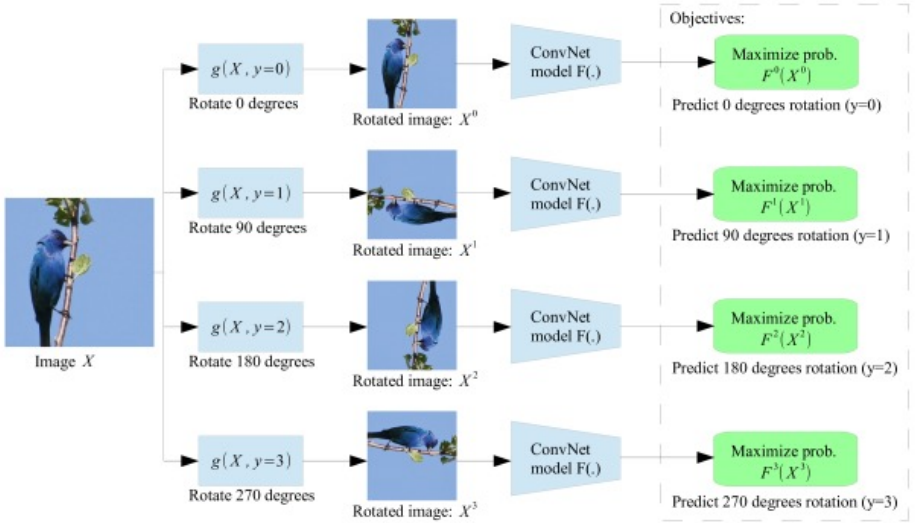
- Learning relative positions of patches helps understanding spatial structures

  [Doersch et al. ICCV 2015]



- Learning rotation angles teaches rotation invariance

  [Gidaris et al. ICLR 2018]



C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in IEEE International Conference on Computer Vision (ICCV), pp. 1422–1430, 2015.
S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in International Conference on Learning Representations (ICLR), 2018.

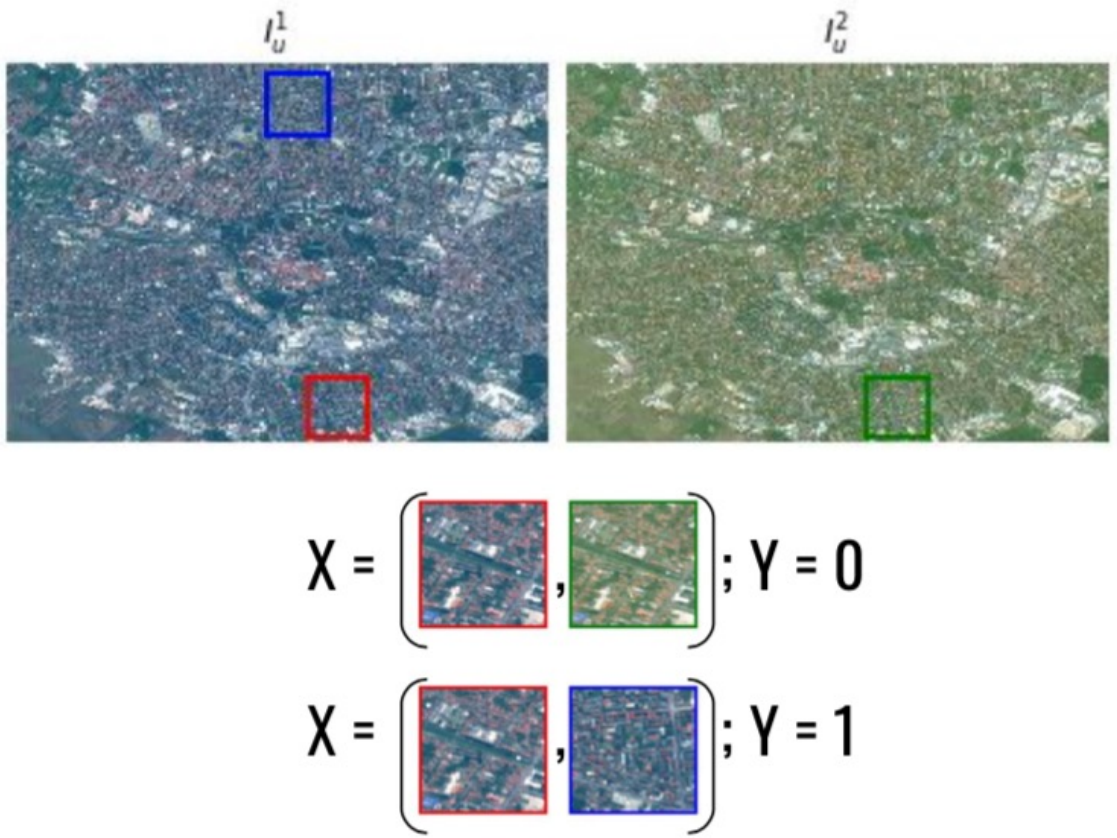
How can this help in change detection?

- We want to warm start a CNN for change detection in an unsupervised way
- We do not like when application irrelevant (e.g. illumination/seasonal) changes get in the way
- In this pair, for example, no changes have occurred

- We want **pretext tasks projecting close samples where no changes have occurred**

Pretext task 1: discriminating overlapping patches

- Binary classification task
  - Red and green patches overlap, y = 0
  - Red and blue do not, y =1



From Leenstra et al., 2021

Pretext task 2 (harder)
push overlapping patches closer in feature space

- We use triplets
  - Red and green overlap
  - Red and blue don't

- Minimize feature distance between red and green

  while

- Maximizing distance between red and blue

From Leenstra et al., 2021

Pretext task 2 (harder)
push overlapping patches closer in feature space

- We use this loss:

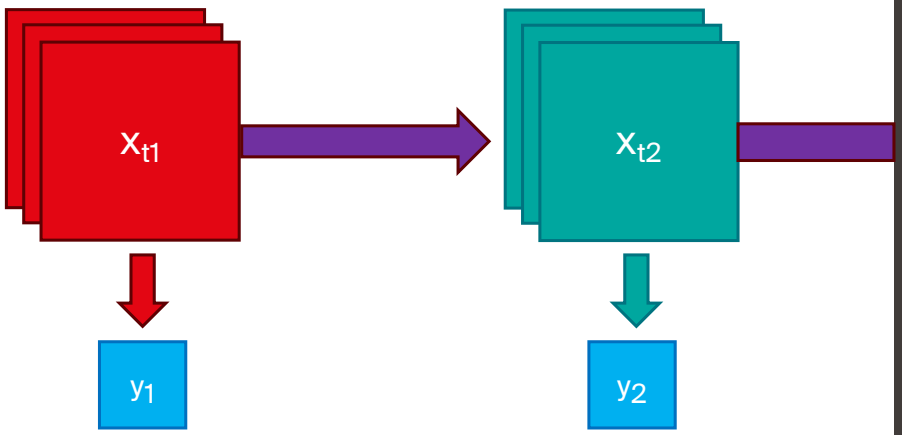
$$L = \frac{1}{RC} \sum_{r=1,c=1}^{R,C} \max(||z^1 - z^2||_2 - ||z^1 - z^3||_2 + m, 0) + \gamma \cdot |z^1 - z^2|$$

- Minimize feature distance between red and green

while

- Maximizing distance between red and blue

From Leenstra et al., 2021



Monitor a changing process with multiple time steps

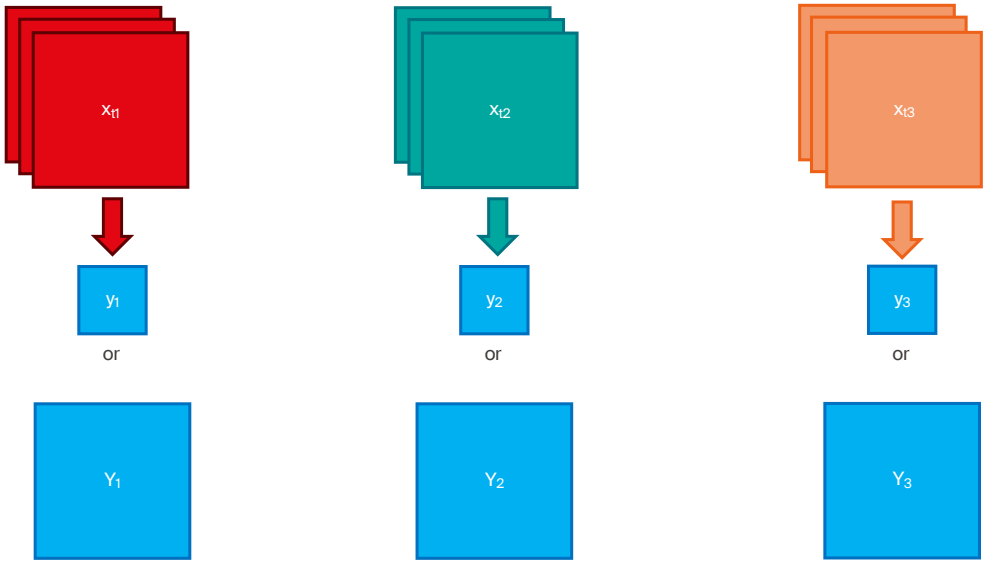
RNNs and LSTMs

It’s all about passing information

Now about sequences

- If we look at image sequences, the classes might vary.
- E.g. in agriculture, soil → crop → soil → other crop
- One class per location is too reductive!

- You could classify independently



- knowing what happened before might help!

Sequences are very well studied in NLP*

*Natural language processing

D. Tuia. ECEO

input/feature #1 input/feature #2 output/label

Thou shalt _____

Sequences are very well studied in NLP*

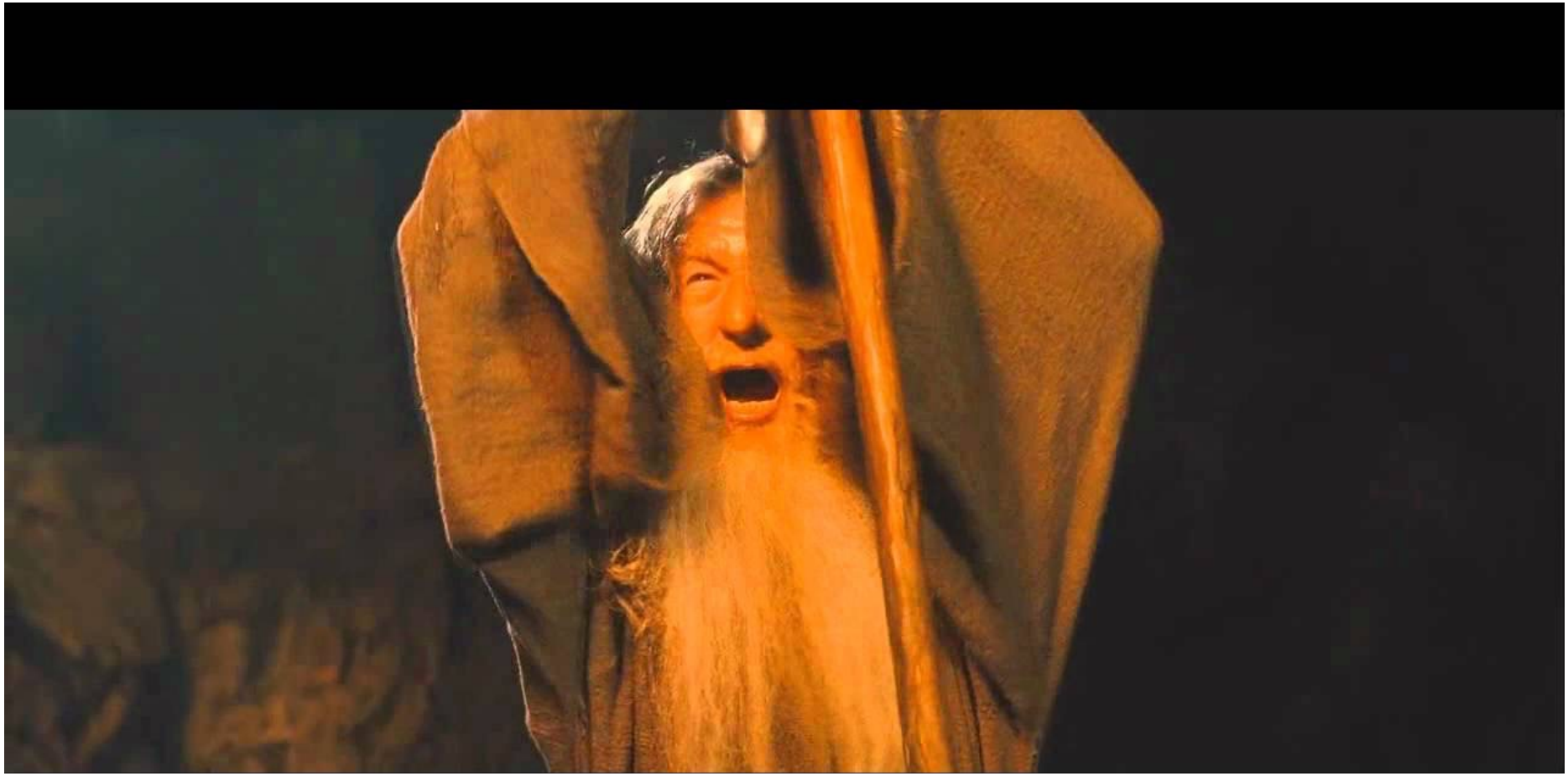
*Natural language processing

D. Tuia. ECEO

input/feature #1 input/feature #2 output/label

You shall _____

Sequences are very well studied in NLP*



Sequences are very well studied in NLP*

*Natural language processing

D. Tuia. ECEO

input/feature #1 input/feature #2 output/label

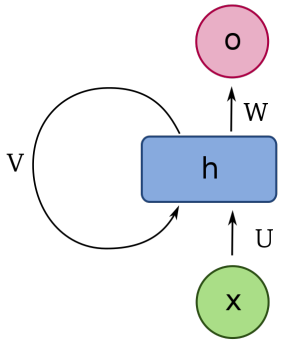
You shall _____

Recurrent neural network

- We will consider a model that applies recursion
- A.k.a, a network that feeds its outputs as inputs for the next time step

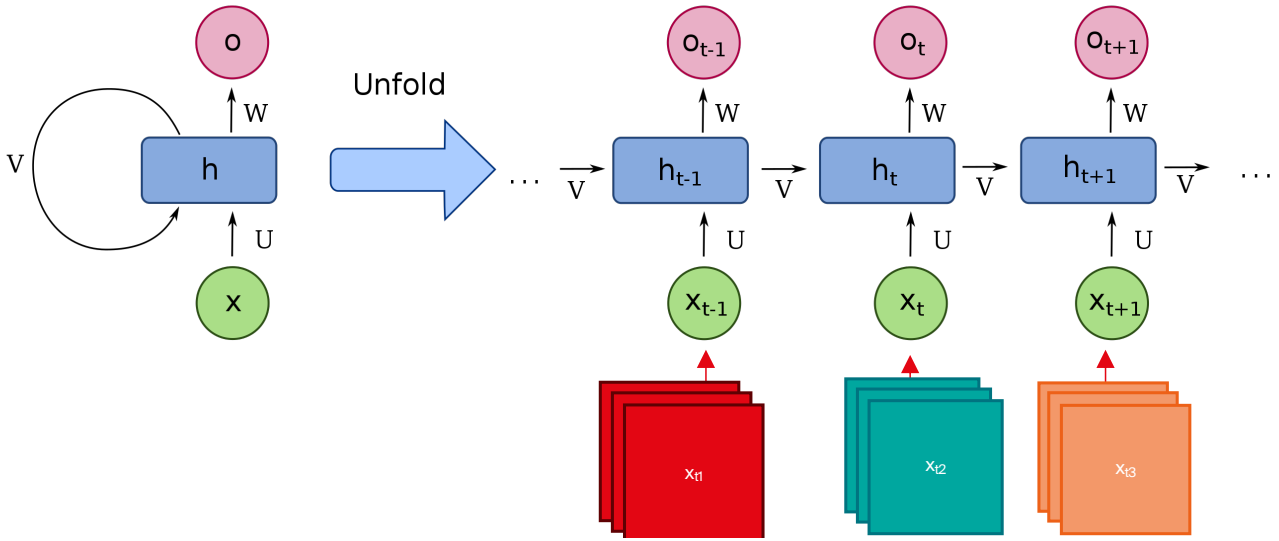
Recurrent models

- This means that
  - The model goes through the sentence one word at a time
  - For each word, it extracts a feature representation
  - It injects such representation into the model considering the next word



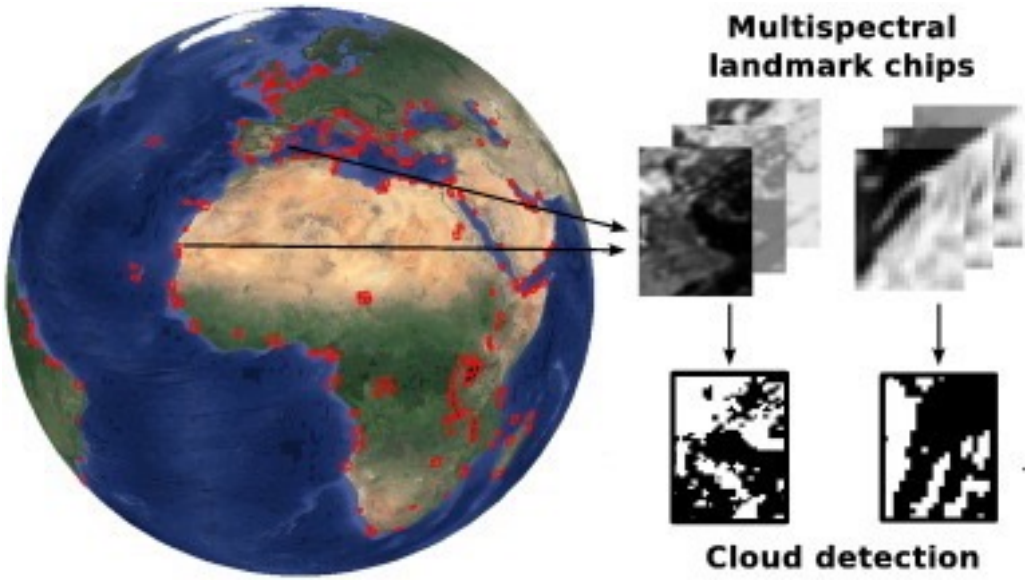
Recurrent models

- This means that
  - The model goes through the sentence one word at a time
  - For each word, it extracts a feature representation (h) with model (U)
  - It injects such representation (v) into the model considering the next word

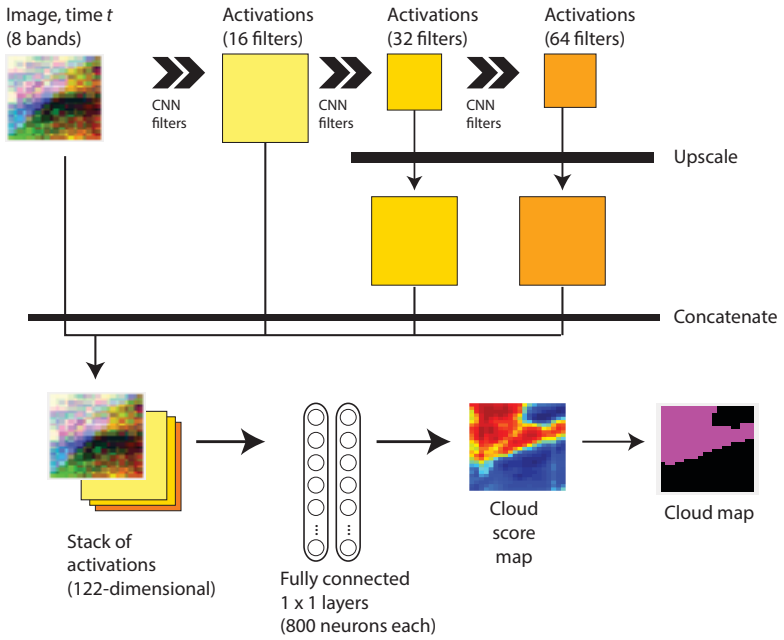


Let's make it more remote-sensing-y

- Task: clouds classification
- Semantic segmentation
- Images of geostationnary satellite
- SEVIRI infrared bands (MeteoSAT)
- 1 location: 35,040 time steps (1 year every 15 minutes)

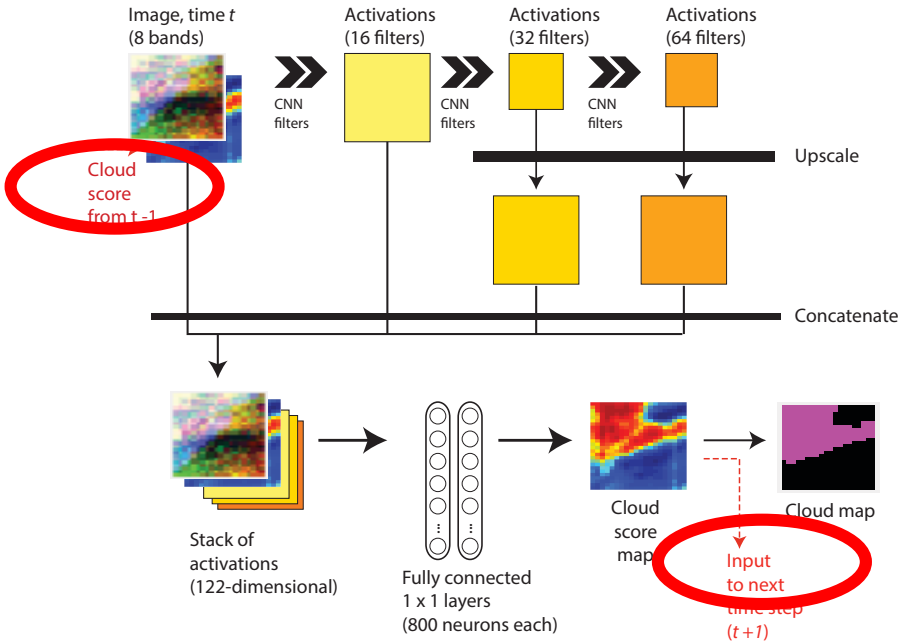


Our base cloud classification CNN: hypercolumn (see last course)



D. Marcos, M. Volpi, B. Kellenberger, and **D. Tuia**. Land cover mapping at very high resolution with rotation equivariant CNNs: towards small yet accurate models. *ISPRS J. Int. Soc. Photo. Remote Sens.*, 145(A):96–107, 2018.

Adding recursion to the hypercolumn

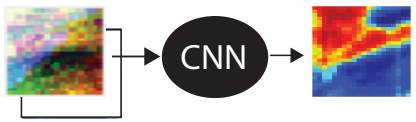


From CNN to recursion

- Design a single timestep CNN (here for semantic segmentation)
- Add as input a (r x c x 2) - dimensional array of zeros
- Copy the network t times
- Declare the (r x c x 2) input of t as the fully connected scores at t-1

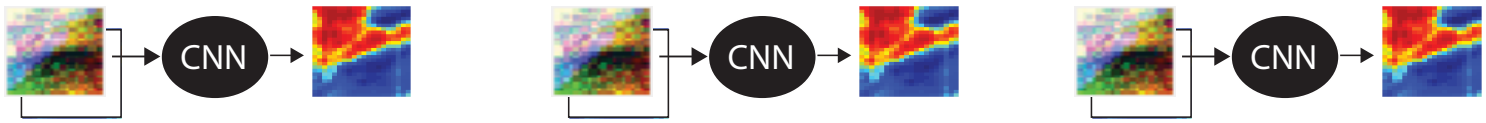
From CNN to recursion

- Design a single timestep CNN
- Add as input a (r x c x 2) - dimensional array of zeros
- Copy the network t times
- Declare the (r x c x 2) input of t as the fully connected scores at t-1



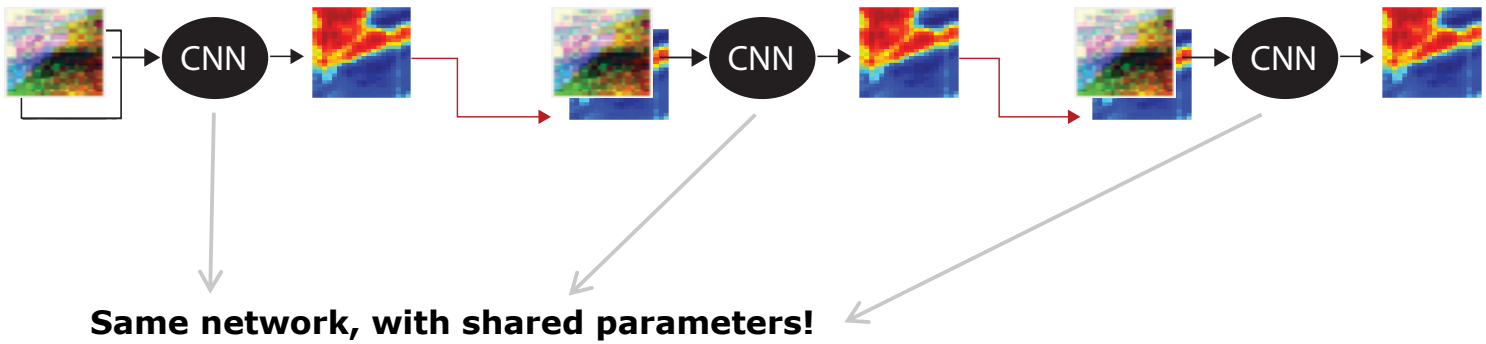
From CNN to recursion

- Design a single timestep CNN
- Add as input a (r x c x 2) - dimensional array of zeros
- Copy the network *t* times
- Declare the (r x c x 2) input of t as the fully connected scores at t-1



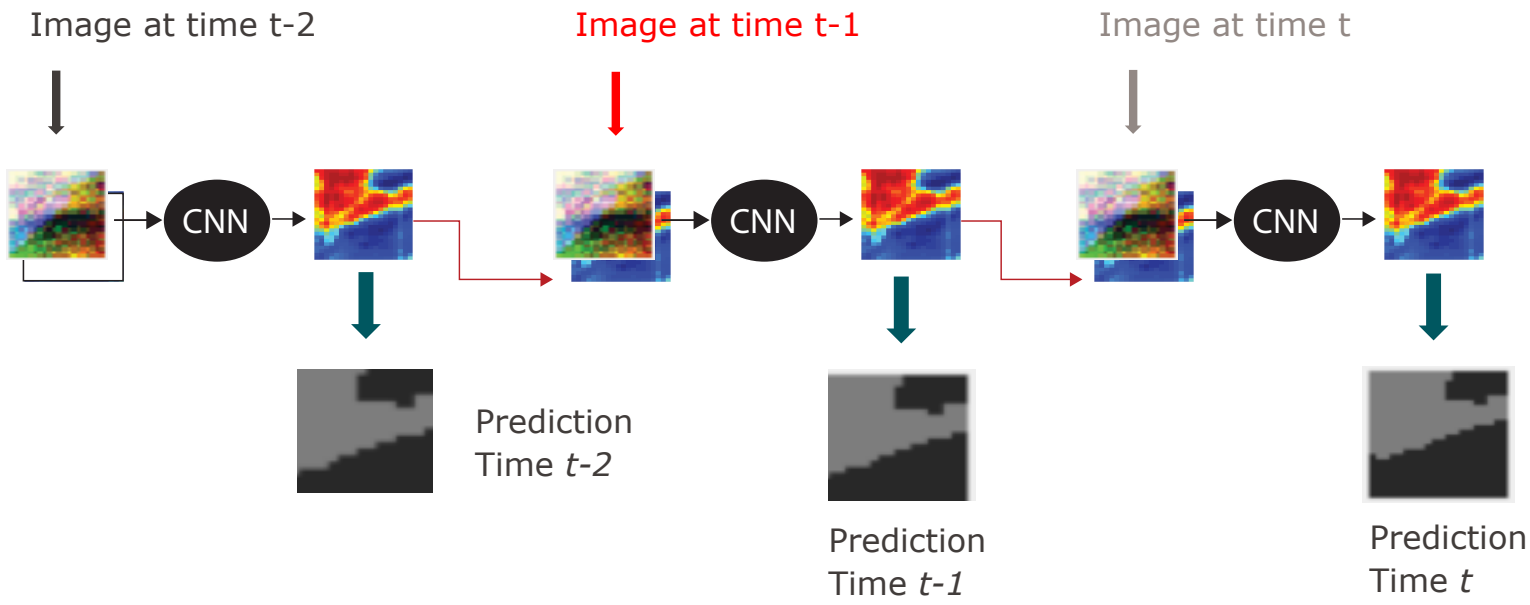
From CNN to recursion

- Design a single timestep CNN
- Add as input a (r x c x 2) - dimensional array of zeros
- Copy the network t times
- Declare the (r x c x 2) input of t as the fully connected scores at t-1



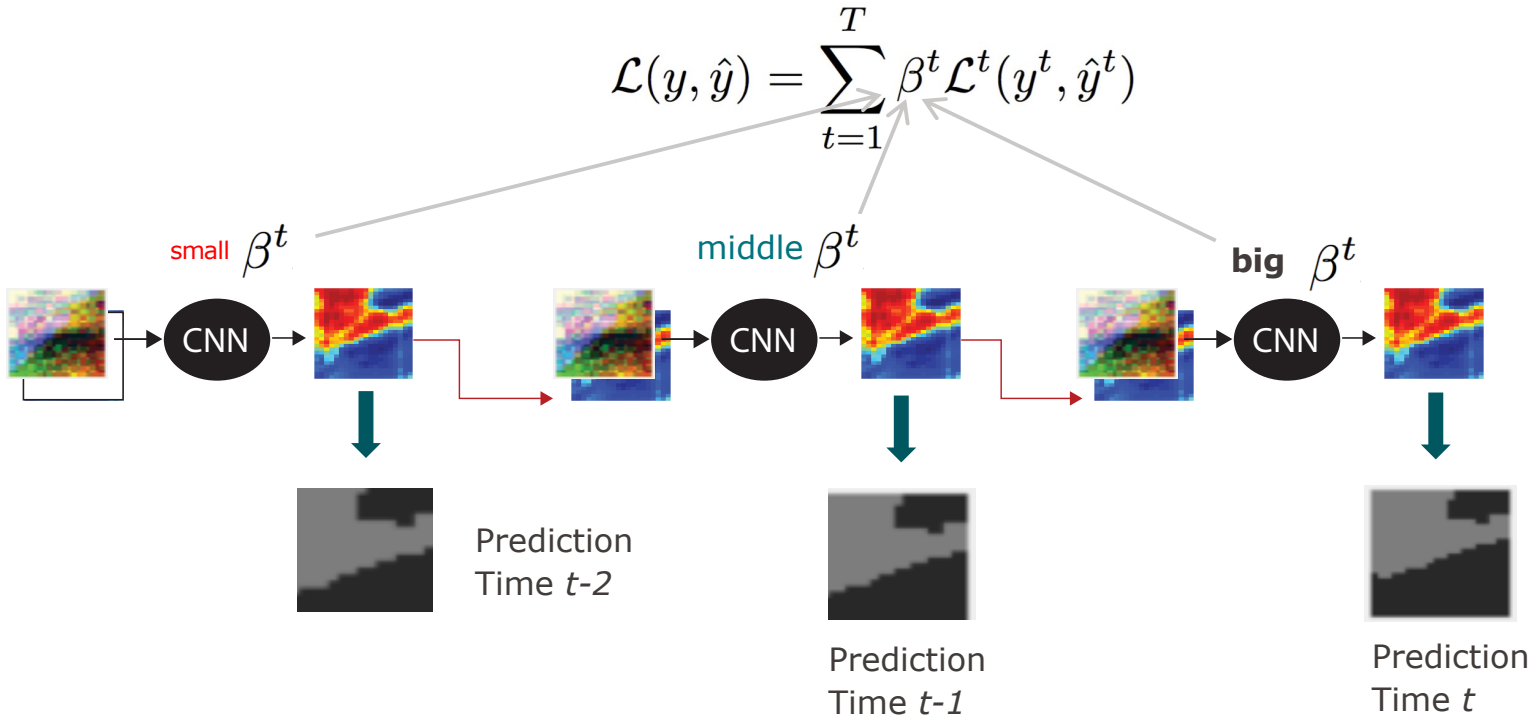
Training the network

- Pass every image time sequence in the network
- Backpropagate through the network



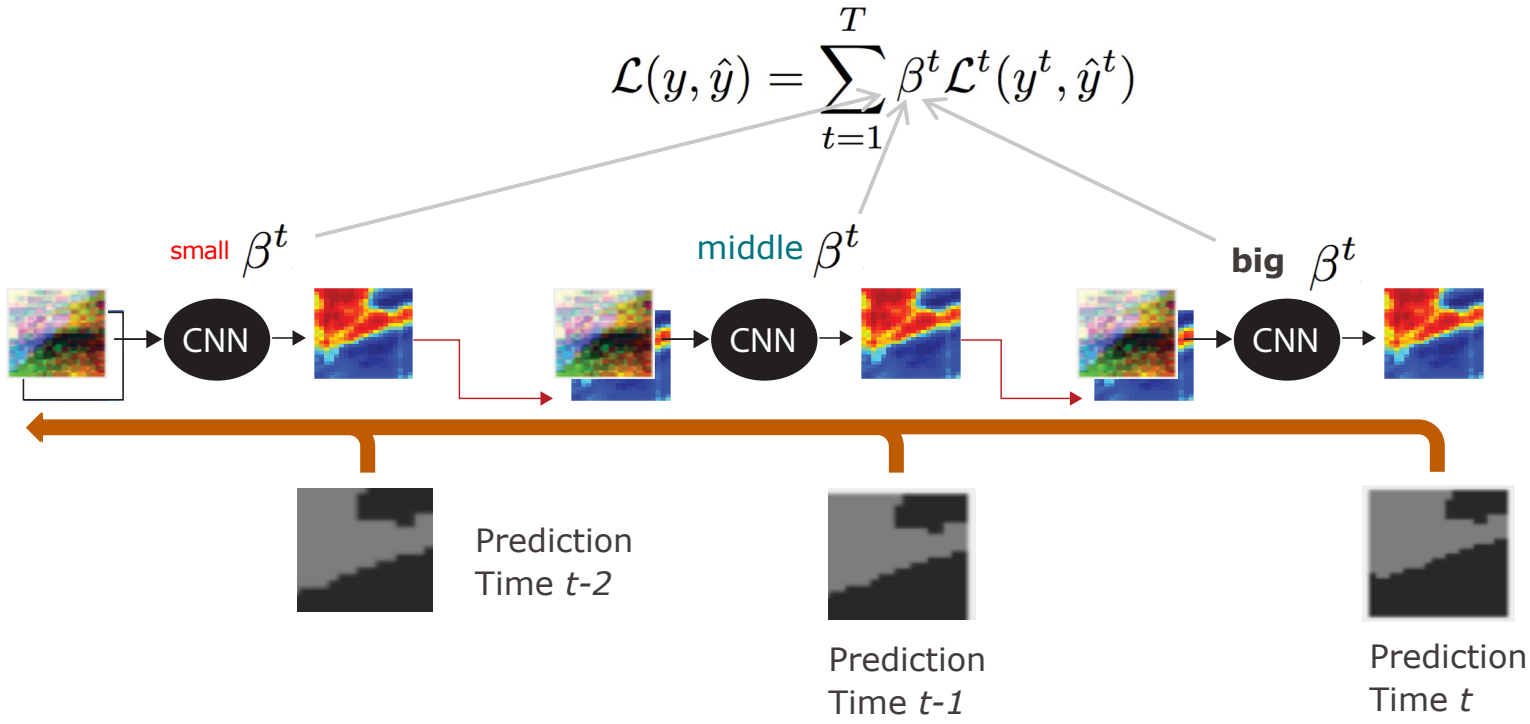
Training the network

- Pass every image time sequence in the network
- Backpropagate through the network, update w in each CNN



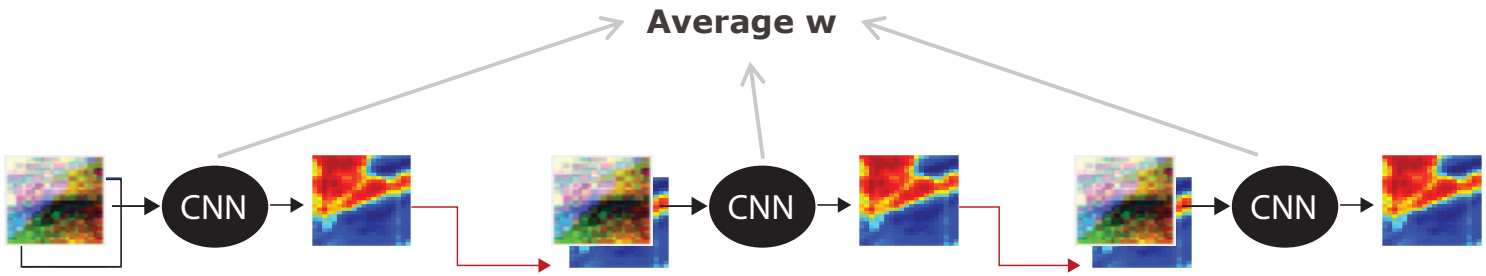
Training the network

- Pass every image time sequence in the network
- Backpropagate through the network, update w in each CNN

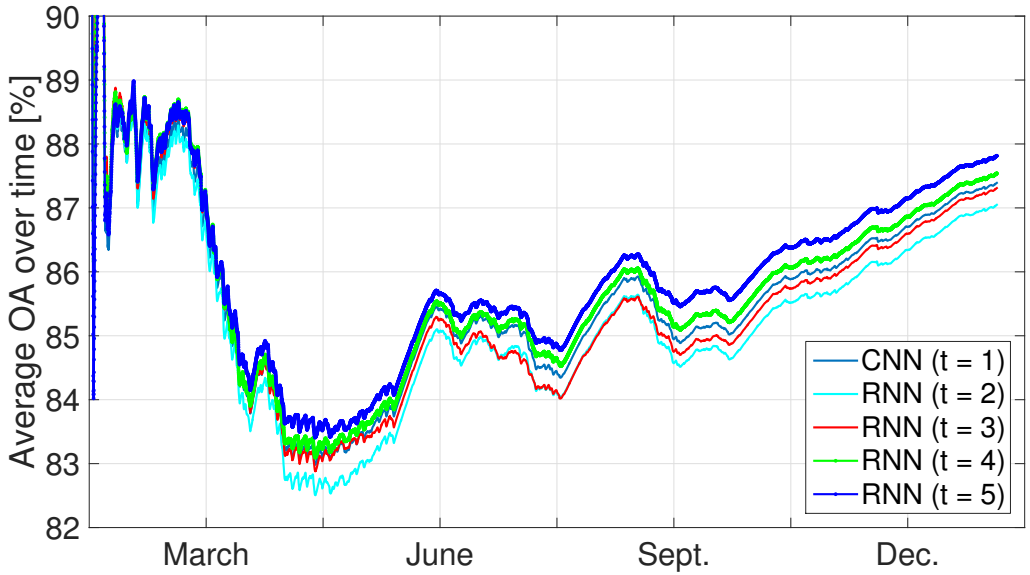


Training the network

- Pass every image time sequence in the network
- Backpropagate through the network
- Average weights



Results



Video

Setting

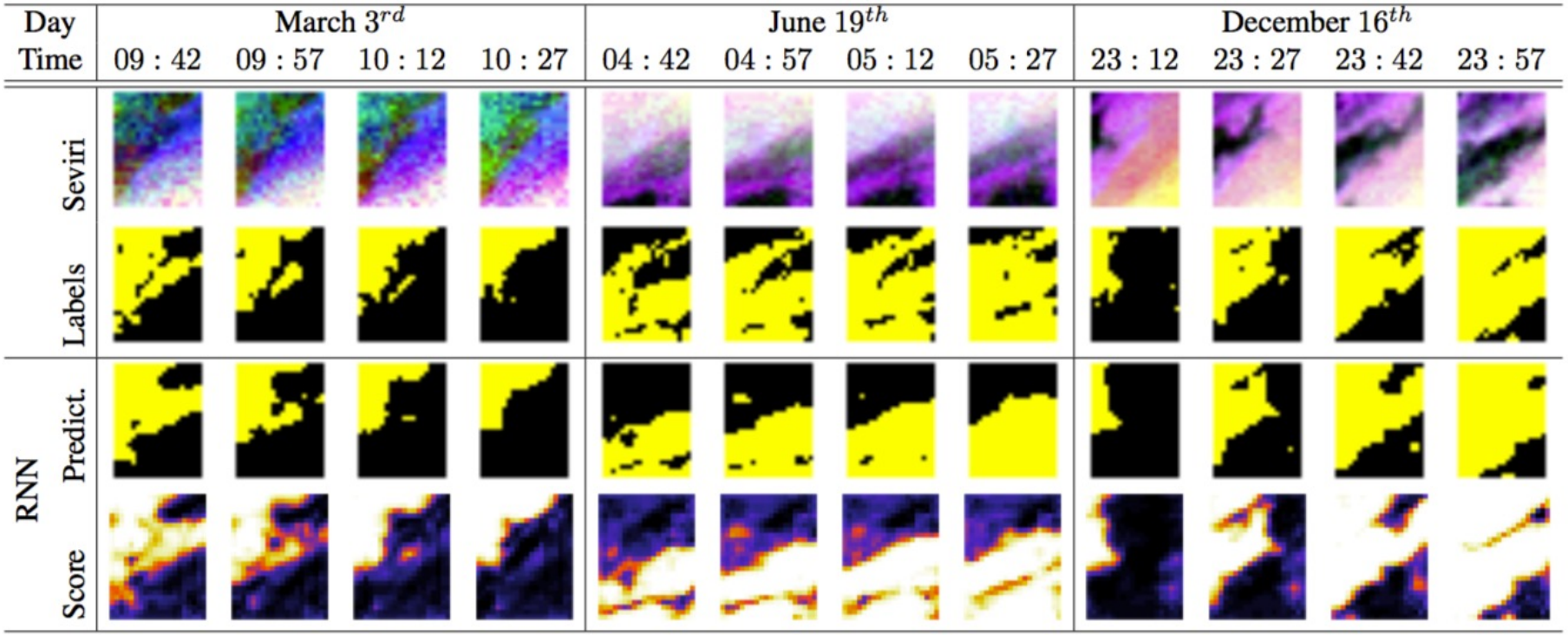
- Test: March, June, September, December
- Train: all other months

- Metric: weighted average accuracy (over timesteps t), so it's normal error increases over time

Take home

- Adding time information includes prediction accuracy
- We get the prediction for all the time sequence
- Consistency is increased

Results


Day
Time
March 3rd
09 : 42
09 : 57
10 : 12
10 : 27
June 19th
04 : 42
04 : 57
05 : 12
05 : 27
December 16th
23 : 12
23 : 27
23 : 42
23 : 57
Seviri
Labels
RNN
Predict.
Score

RNNs in a nutshell

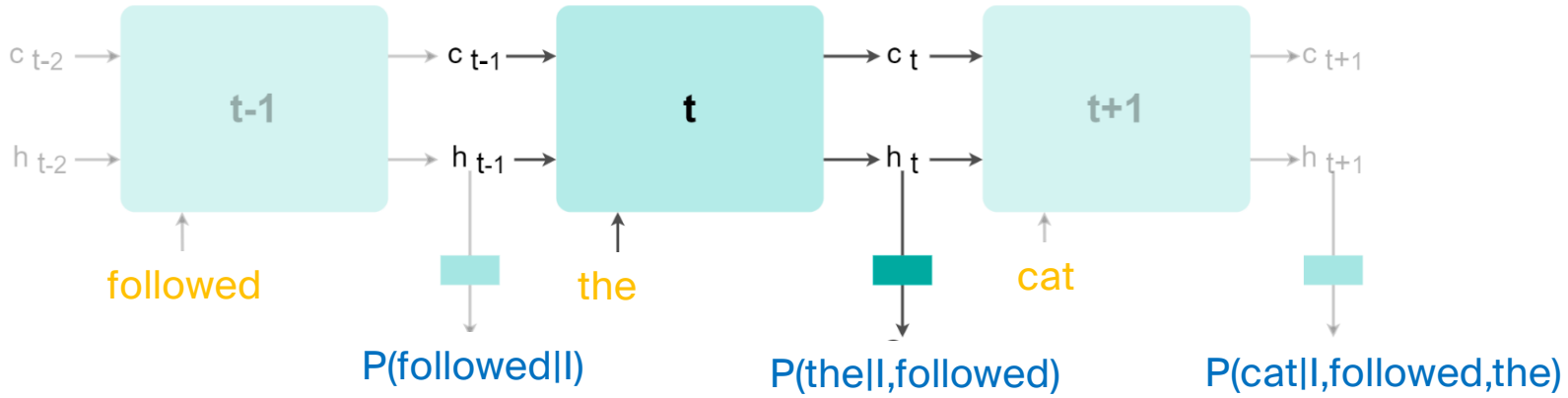
- Recursive model that can be unrolled
- Use t-2 output as input to t-1 and so on
- Very intuitive and works well!

But:

- Does not handle well very long sequences
- Because gradients are pushed through the network from end to beginning
- To alleviate this, Hochreiter and Schmidhuber (1997) proposed a solution, which became state of art: the **L**ong-**S**hort **T**erm **M**emory network (LSTM)

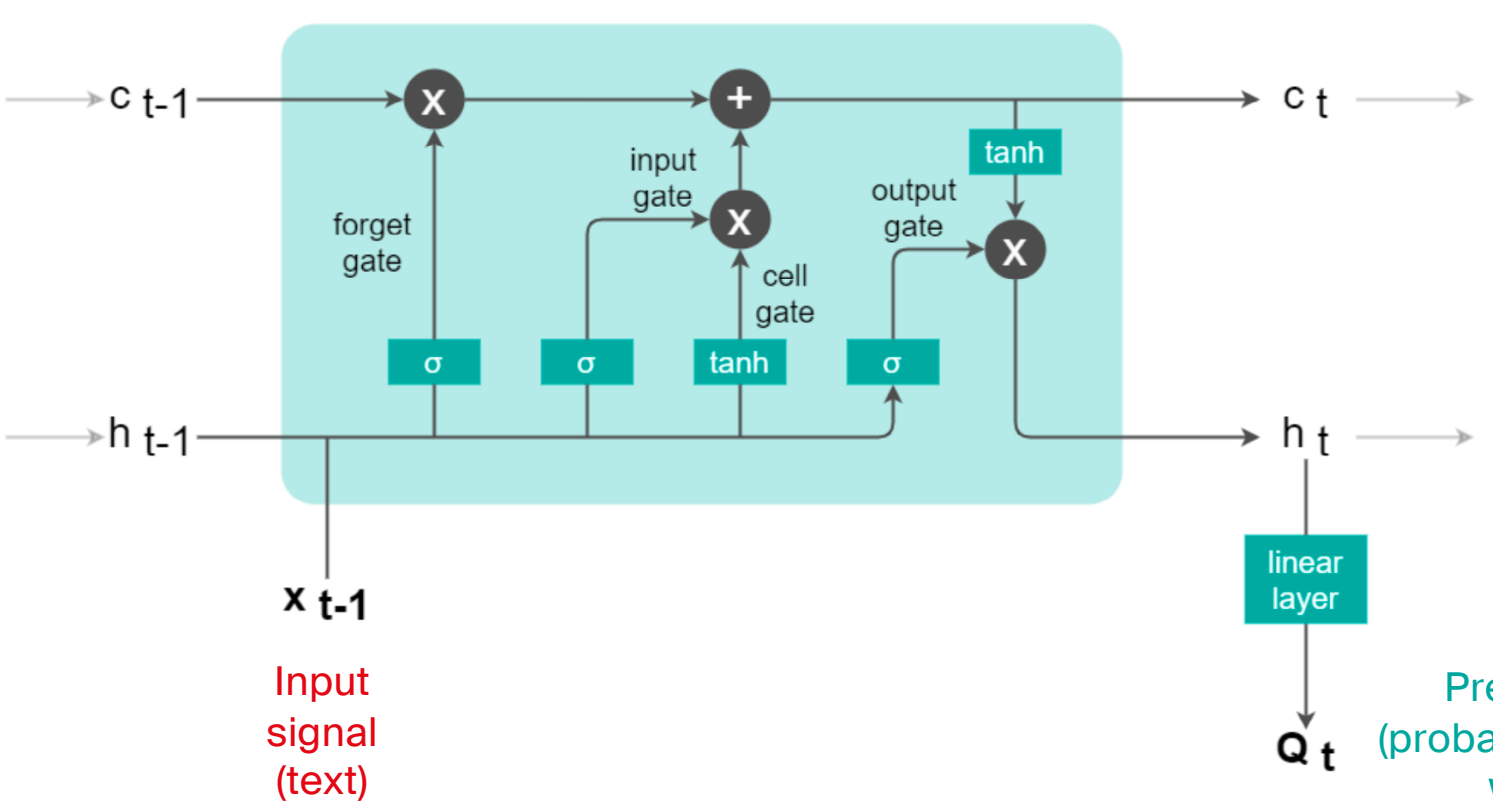
LSTM (unrolled view)

- Like a RNN, the LSTM considers one word at a time in the sentence direction;
- Like a RNN, the LSTM passes on a hidden vector *h* to the next step*;*
- *Unlike* a RNN, the LSTM also passes a **long memory** vector *c*.



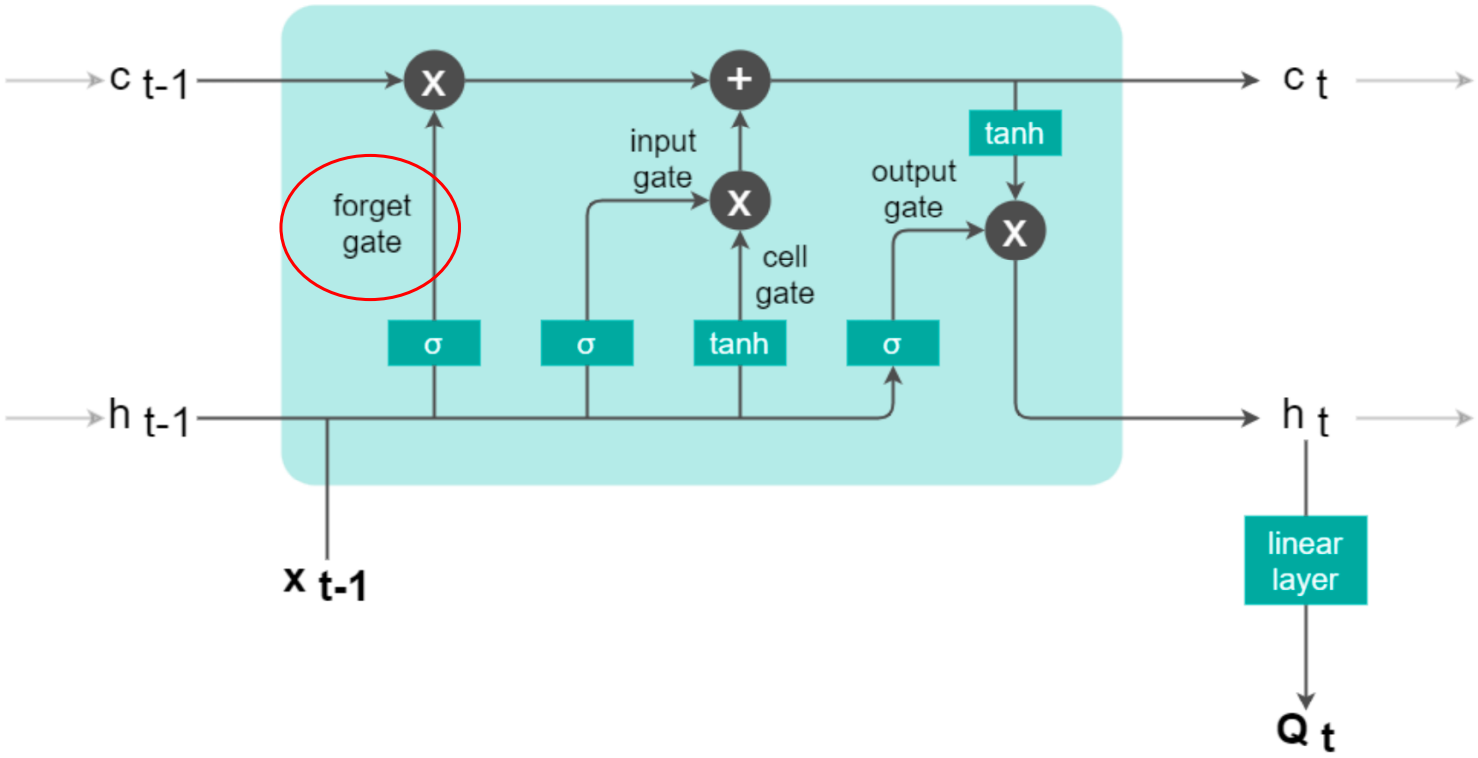
C. Frank, M. Russwurm, J. Fluixa, A. Abellan, and D. Tuia. Short-term runoff forecasting in an alpine catchment with a long short-term memory neural network. *Frontiers in Water and AI*, 5, 2023. https://www.frontiersin.org/journals/water/articles/10.3389/frwa.2023.1126310/full

The LSTM cell



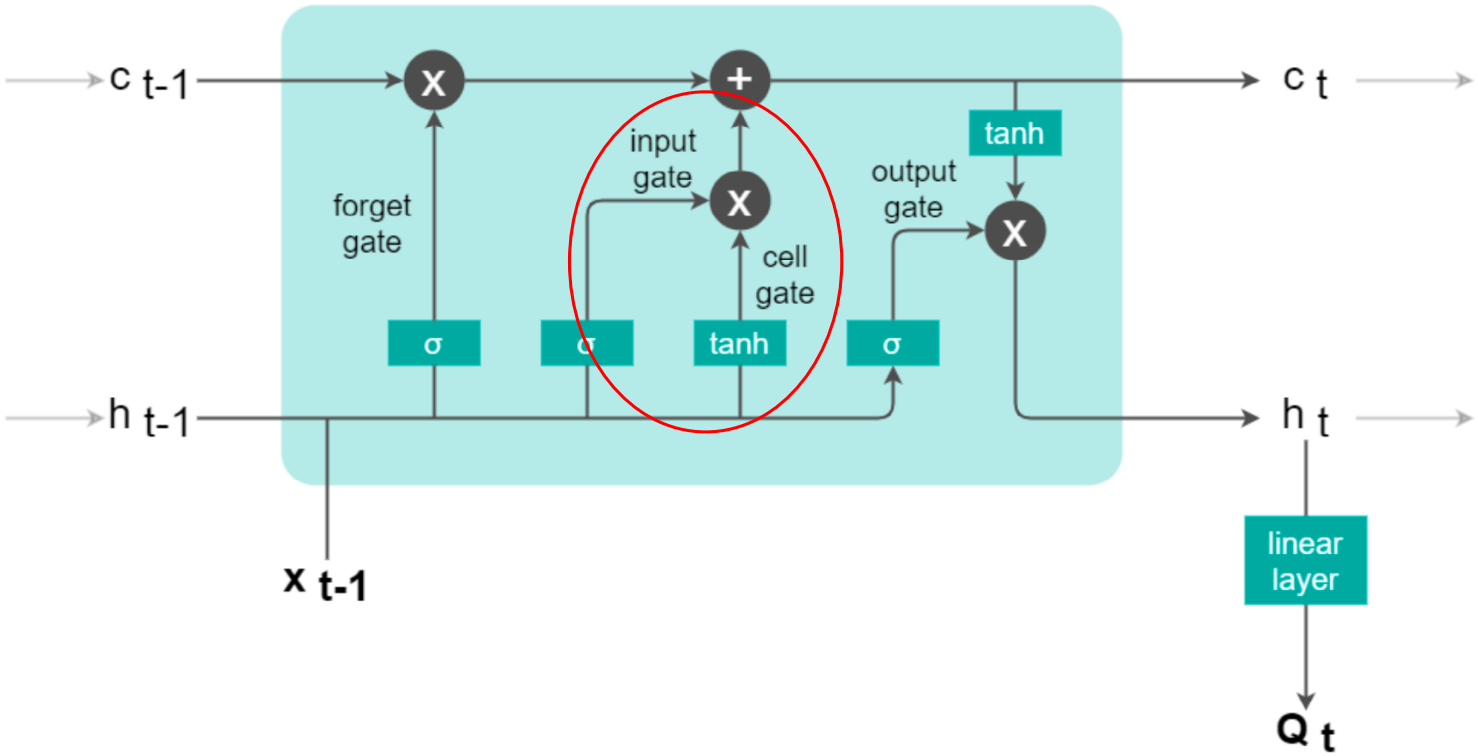
C. Frank, M. Russwurm, J. Fluixa, A. Abellan, and D. Tuia. Short-term runoff forecasting in an alpine catchment with a long short-term memory neural network. *Frontiers in Water and AI*, 5, 2023. https://www.frontiersin.org/journals/water/articles/10.3389/frwa.2023.1126310/full

The LSTM cell



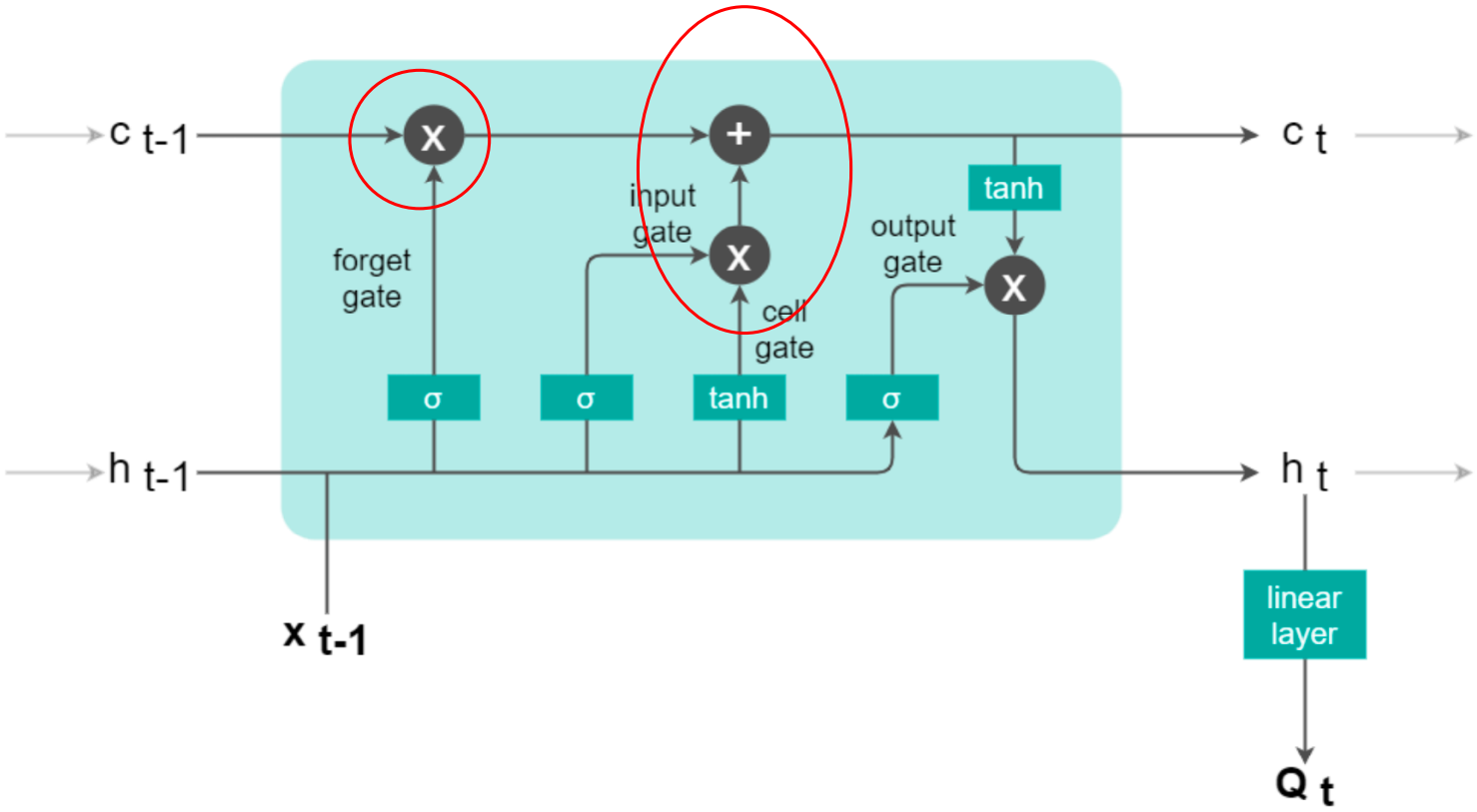
The forget gate regulates how much of the info in the long term memory c is deleted from the memory

The LSTM cell



The input and cell gate regulate how much of the inputs is to be added in the long term memory

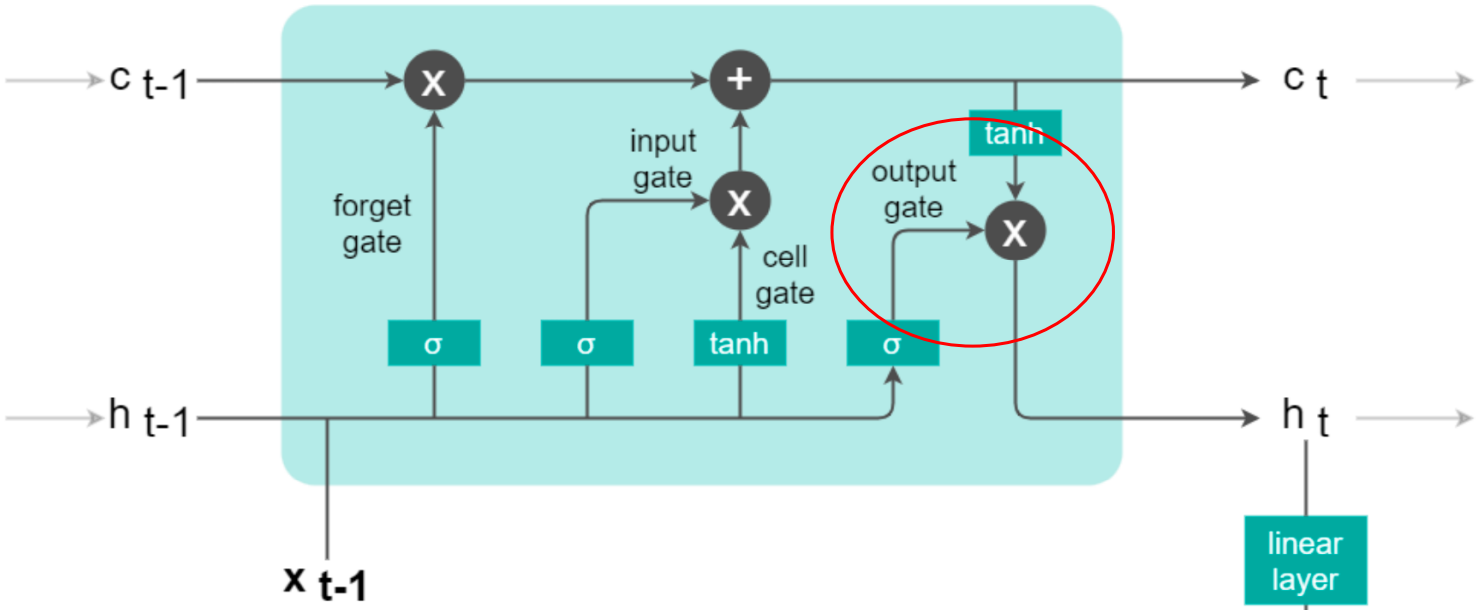
The LSTM cell



Together, they update the long-term memory c:

$$c_{t+1} = \underbrace{f_{t+1} \cdot c_t}_{\text{forget}} + \underbrace{i_{t+1} \cdot g_{t+1}}_{\text{input}}$$

The LSTM cell



The output gate combines the inputs (x and h), modulated by the long term memory c to obtain the new hidden state h

$$h_{t+1} = o_{t+1} \cdot \tanh\left(c_{t+1}\right).$$

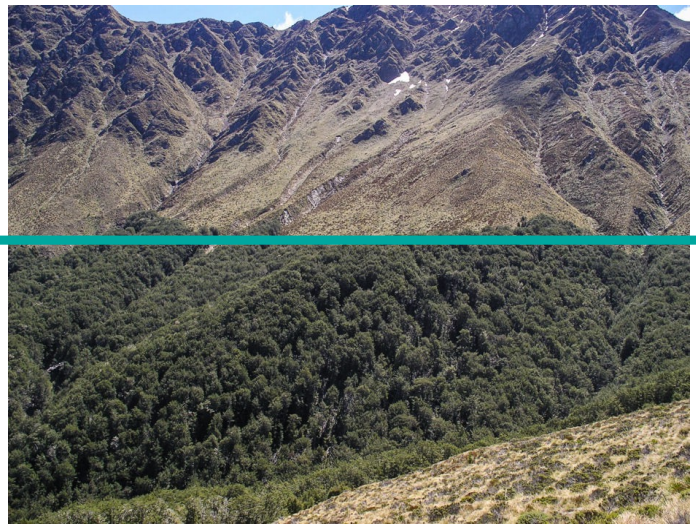
In a nutshell

- The LSTM provides a mechanism to pass information across the sequence step (the long-term memory c)
  1. Go one word at a time
  2. Update *c*
  3. Estimate hidden state for the current word using new input *x*, previous states *c* and *h*
  4. Use *h* to predict P(word | previous_ones)
  5. Pass *h* and *c* on to the next word in the sequence

- This means information from multiple steps ago gets injected directly!
- For the rest it looks pretty much like the RNN.
- All the gates computations involve weight matrices with learnable parameters (every time x_t or h_t are used)

A last example

- We wanted to map the upper limit of the forest in Valais
- This limits moves in time, due to climate change and landuse practices
- We have 80 years of historical images from Swisstopo
- But labels only in 2020!
- We developed a sequence-based approach



https://farsouthecology.com/do-treelines-in-the-southern-hemisphere-follow-the-rules/

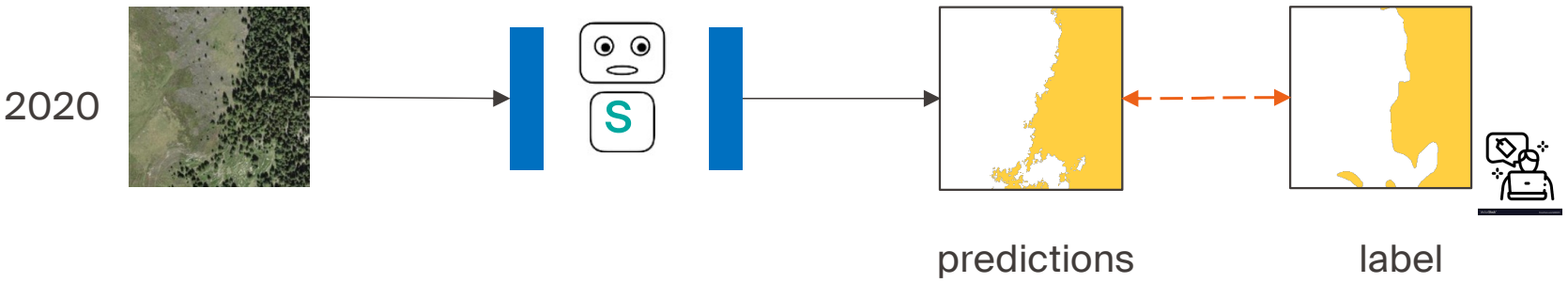
It’s a 80 years adventure with varying technology...

VS

And it shows on the images!

A single time classifier

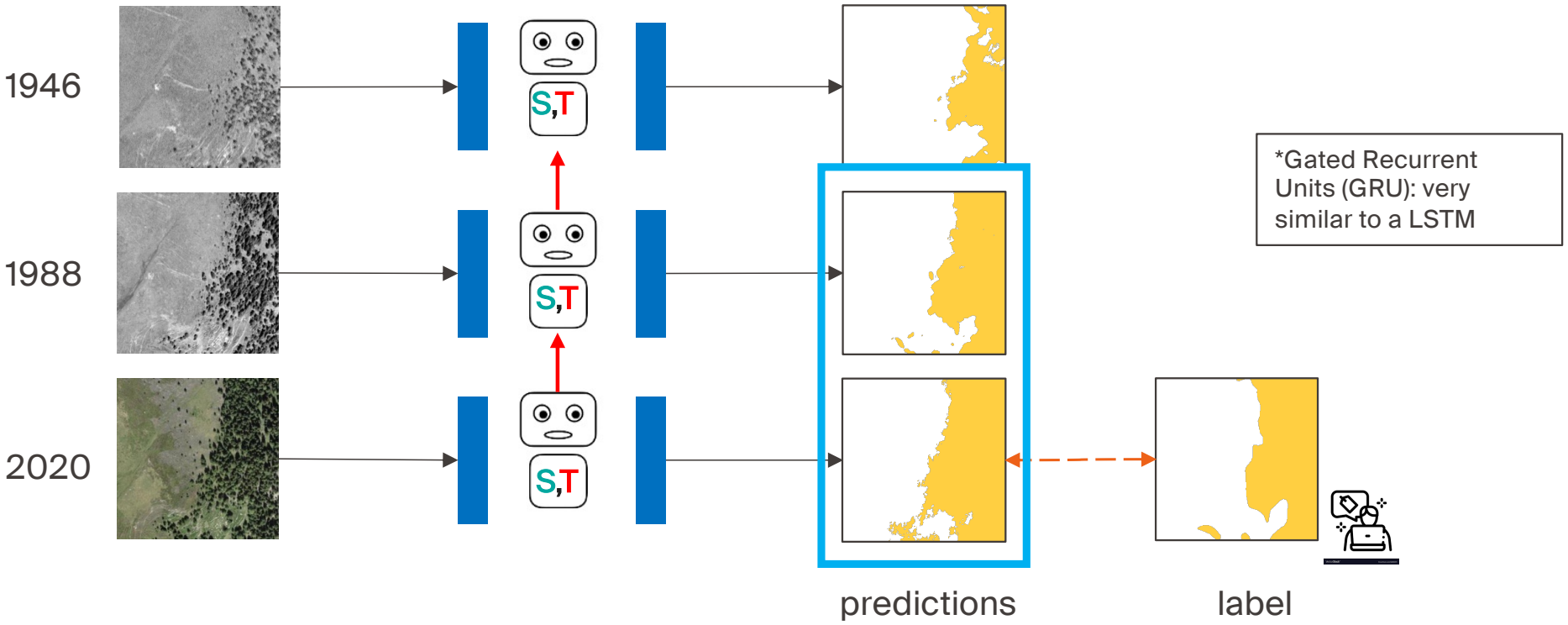
Model: U-Net
Loss: crossentropy between labels and predictions in 2020



T.-A. Nguyen, B. Kellenberger, and D. Tuia. Mapping forest in the Swiss Alps treeline ecotone with explainable deep learning. *Remote Sens. Environ.*, 281(113217), 2022. https://doi.org/10.1016/j.rse.2022.113217

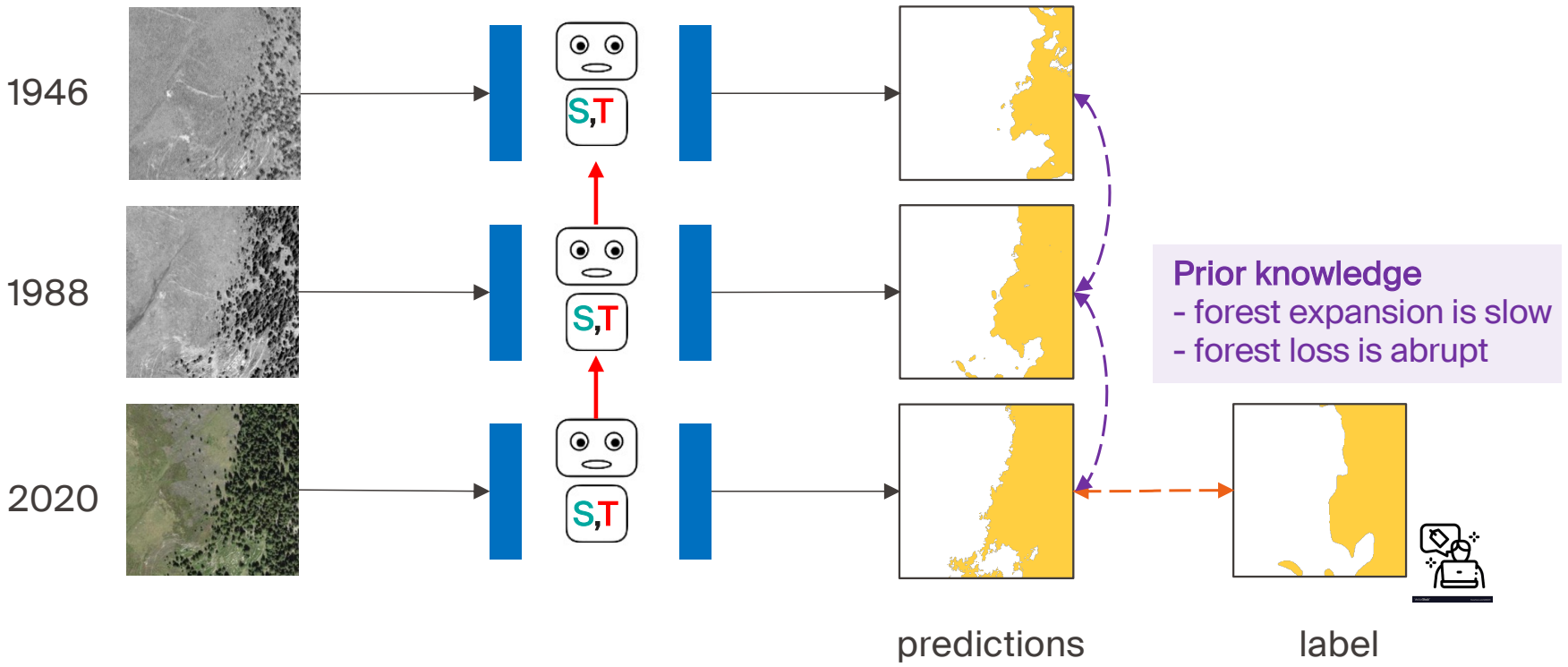
A multitemporal classifier

Model: convolutional GRU*. Gates weighed proportional to time-lag
Loss: as before + cross-entropy between the maps at *t* and *t-1*, *t-1* and *t-2*, etc.



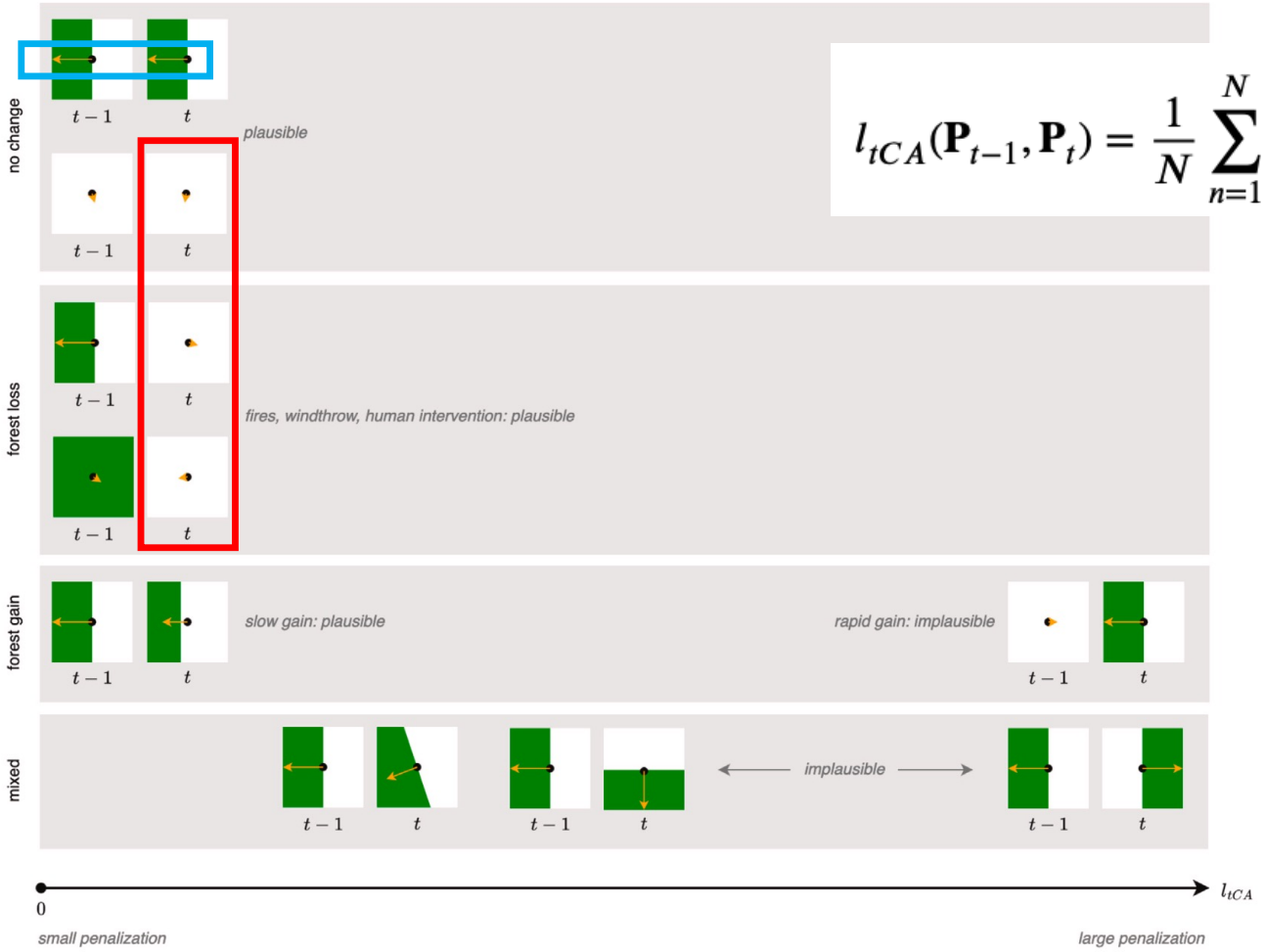
T.-A. Nguyen, M. Russwurm, G. Lenczner, and D. Tuia. Multi-temporal forest monitoring in the Swiss Alps with knowledge-guided deep learning. *Remote Sens. Environ.*, 305:114109, 2024. https://doi.org/10.1016/j.rse.2024.114109

A multitemporal classifier enforcing forest dynamic knowledge



T.-A. Nguyen, M. Russwurm, G. Lenczner, and D. Tuia. Multi-temporal forest monitoring in the Swiss Alps with knowledge-guided deep learning. *Remote Sens. Environ.*, 305:114109, 2024.
https://doi.org/10.1016/j.rse.2024.114109

Temporal evolution loss

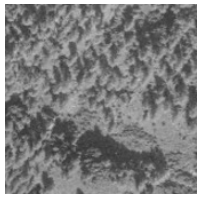


$$l_{tCA}(\mathbf{P}_{t-1}, \mathbf{P}_t) = \frac{1}{N} \sum_{n=1}^{N} \|\nabla p_{n,t}\|_2 \times l_{cos}(\nabla p_{n,t-1}, \nabla p_{n,t})$$

If forest disappears, Gradient at time t is 0.

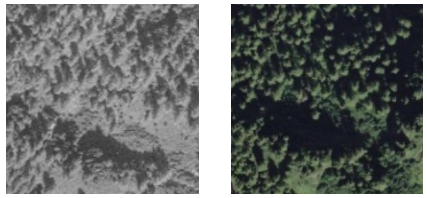
Similarity of gradients between time steps

Numerically



Gray images

Color images



All

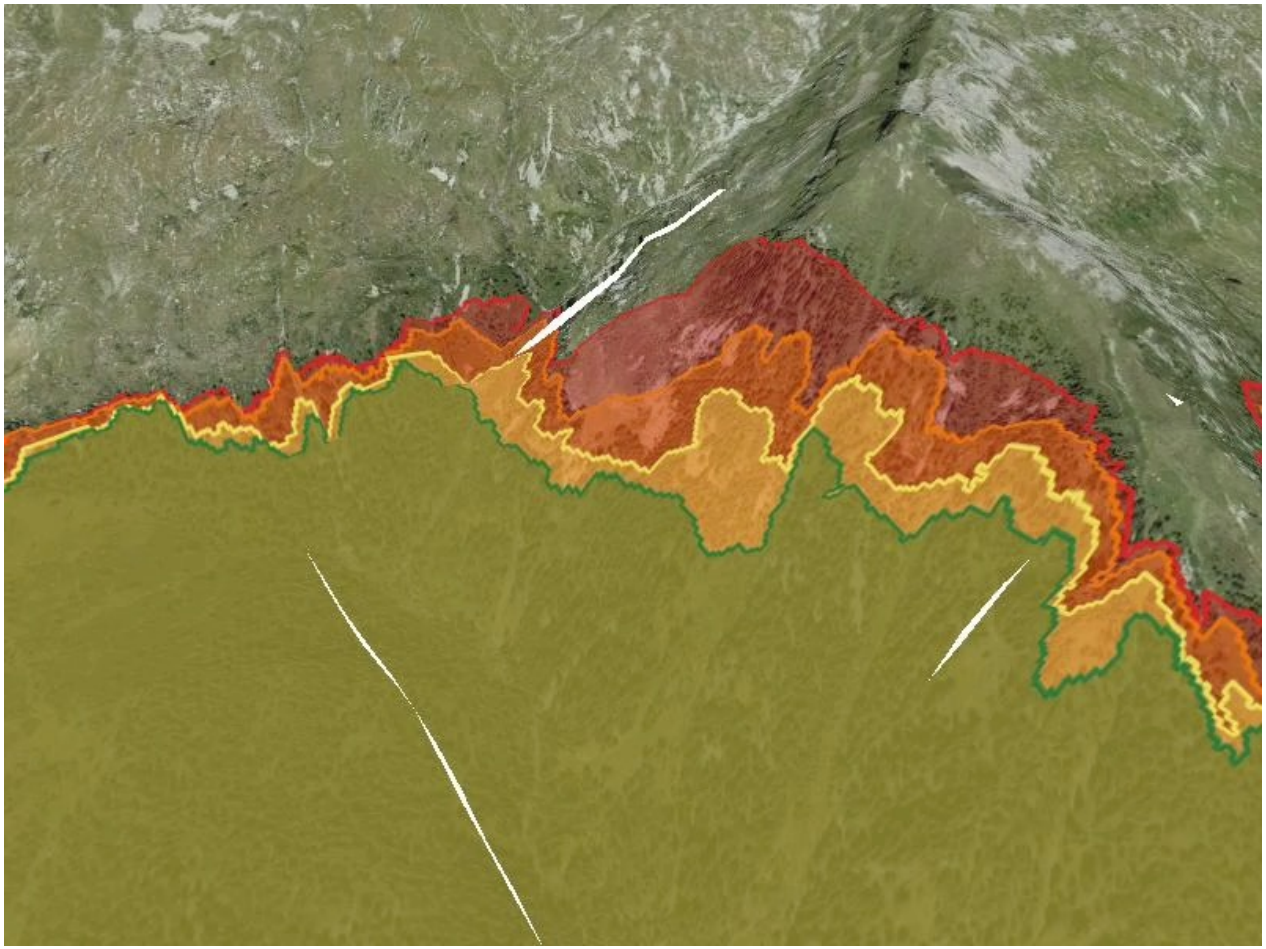
	Gray (1946–1995)			RGB (1998–2020)			All (1946–2020)		
	IoU	F1	$F1_c$	IoU	F1	$F1_c$	IoU	F1	$F1_c$
U-Net	73.5 ± 1.4	84.7 ± 0.9	72.1 ± 2.4	**87.5 ± 0.4**	**93.3 ± 0.2**	81.6 ± 0.3	79.1 ± 0.7	88.4 ± 0.4	75.5 ± 1.6
U-Net + ConvGRU	**76.9 ± 1.9**	**86.9 ± 1.2**	80.1 ± 0.8	85.5 ± 0.3	92.2 ± 0.2	**81.5 ± 0.2**	**80.2 ± 1.3**	**89.0 ± 0.8**	80.6 ± 0.6
U-Net + IrregConvGRU	76.0 ± 1.2	86.3 ± 0.8	**81.0 ± 0.5**	83.0 ± 1.9	90.7 ± 1.1	80.6 ± 0.8	78.7 ± 1.4	88.1 ± 0.8	**80.9 ± 0.3**

Time dimension helps (mostly on historical gray images)!

	Gray (1946–1995)			RGB (1998–2020)			All (1946–2020)		
	IoU	F1	$F1_c$	IoU	F1	$F1_c$	IoU	F1	$F1_c$
tMSE	67.8 ± 2.0	80.8 ± 1.4	64.7 ± 3.2	82.2 ± 8.6	90.0 ± 5.6	76.8 ± 3.3	73.6 ± 4.5	84.7 ± 3.1	69.2 ± 1.6
tCE (Saha et al., 2020)	72.7 ± 1.2	84.2 ± 0.8	69.3 ± 2.0	**87.2 ± 0.4**	**93.2 ± 0.2**	80.3 ± 0.7	78.5 ± 0.8	88.0 ± 0.5	73.3 ± 1.5
tCA	68.5 ± 4.9	81.2 ± 3.5	78.2 ± 1.4	73.7 ± 7.2	84.6 ± 4.8	77.2 ± 1.8	70.5 ± 5.5	82.6 ± 3.8	77.9 ± 1.3
tMSE + tCA	73.2 ± 2.7	84.5 ± 1.8	76.6 ± 2.0	80.4 ± 6.8	89.0 ± 4.4	78.7 ± 2.0	76.0 ± 4.1	86.3 ± 2.7	77.3 ± 1.3
tCE + tCA	**76.0 ± 1.2**	**86.3 ± 0.8**	**81.0 ± 0.5**	83.0 ± 1.9	90.7 ± 1.1	**80.6 ± 0.8**	**78.7 ± 1.4**	**88.1 ± 0.8**	**80.9 ± 0.3**

Forest dynamics knowledge helps!

Mapping Swiss forests



In summary

- Time is a goldmine in remote sensing
- Models exist to account for the time dimension in remote sensing
- We have seen 3 cases where time plays a more and more important role
  - Classification with temporal inputs
  - Change detection
  - Sequence steps classification