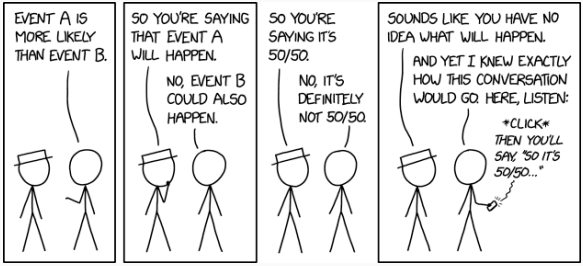
Paper club: Multi-time attention networks for irregulary sampled time series

Julien MICHEL, Iris DUMEUR (and Valentine, Jordi, Mathieu, Silvia, Yoel, Juan . . .)

DS@CB - Novermber 8th 2022

CESBIO, **Université de Toulouse**, **CNES/CNRS/INRAe/IRD/UPS**, **Toulouse**, **FRANCE**

https://xkcd.com/2451/



https://xkcd.com/2370/

Introduction

Why we need algorithms that can handle irregular sampling in SITS

SITS are irregularly sampled

- Clouds for optical modality
- Number of orbits covering a location may vary
- Satellites may be temporarily unavailable
- We may want to mix series from different missions, each with its own sampling

Machine Learning often expect regular sampling and are not coordinates aware (being time or others)

- Because we keep data in arrays and tensors
- Many of the underlying math expect regular sampling (for ex. convolution) or are blind wrt. sampling

The usual workaround is re-sampling and smoothing

- Temporal re-sampling implies implicit priors about signal and space to store re-sampled data
- Smoothing can obliterate interesting features in signal
- Recent Neural Networks architecture (e.g. Transformers) mix positional encoding and self-attention

Luckily for us, this problem is not limited to Satellite Time Series ...

[1] Shukla,·S. N., & Marlin, B. M. (2021).Multi-time attention networks for irregularly sampled time series. arXiv preprint 2101.10318 (paper, code)

[2] Shukla, S. N., & Marlin, B. M. (2021). Heteroscedastic Temporal Variational Autoencoder For Irregularly Sampled Time Series. arXiv preprint 2107.11350 (paper, code)

"This work is motivated by the analysis of physiological time series data in electronic health records, which are sparse, irregularly sampled, and multivariate."

"In this work, we introduce a new model for multivariate, sparse and irregularly sampled time series that we refer to as Multi-Time Attention networks or mTANs."

"The encoder takes the irregularly sampled time series as input and produces a fixed-length latent representation over a set of reference points, while the decoder uses the latent representations to produce reconstructions conditioned on the set of observed time points."

Aim of this talk

Understand how mTAN and HET Variational AutoEncoders work and what they really do

- Papers not very easy to understand, code provided but hard to read

```
query, key = [l(x).view(x.size(0), -1, self.h, self.embed_time_k).transpose(1, 2) for l, x in zip(self.linears,(query, key))]
```

- Discrepancies between code and papers
- $\Rightarrow$ Code completely rewritten

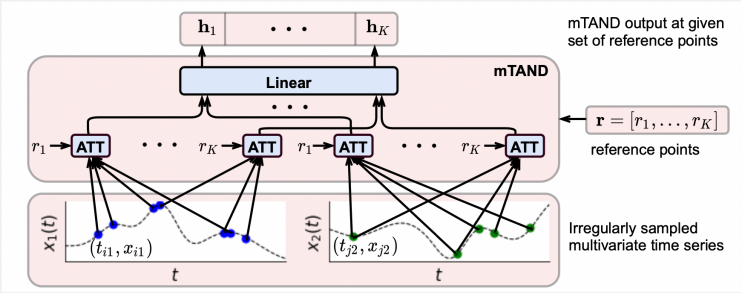
Illustrate how those networks perform on real SITS data

- We harvested multi-modal (S1 + S2) time-series from the PASTIS-R dataset
- Demonstration of the proposed architectures for S1 guided interpolation of S2 NDVI Time Series

Assess usefulness and limitations

mTAN and unTAN time encoder modules explained

mTAN overview (figure from [1])



1. Compute observed times embeddings [L, E] and reference time [K, E]
2. Compute learnable attention scores matrix from reference to observed [L,K]
3. Interpolate input signal with learned kernels (rows of the attention matrix)

Learnable time-embedding

Definition

- Learnable positional encoding embeds a time point into a d_r dimensional space
- Each feature i has its own learned periodic function, ω and β area learned parameters

$$\phi(t)[i] = \begin{cases} \omega_0 t + \beta_0 & \text{if} i = 0 \\ sin(\omega_i t + \beta_i) & \text{if} 1 \leq i \leq d_e \end{cases}$$

Code

```
self.periodic_time_layer = torch.nn.Linear(1, self.full_time_embedding_dim - 1)
self.linear_time_layer = torch.nn.Linear(1, 1)
...
linear_embedding = self.linear_time_layer(time_points) # i=0
periodic_embedding = torch.sin(self.periodic_time_layer(time_points)) # 1<i<d_e
return torch.cat([linear_embedding, periodic_embedding], -1)
```

Time-driven attention scores

In the paper

$$k(r,t) = \frac{exp(\phi(r) w v^T \phi(t)^T \sqrt{d_K})}{\Sigma_{t_i=1}^{L} exp(\phi(r w v^T \phi(t_i)^T / \sqrt{d_K})}$$

In the code

```
self.W = torch.nn.Linear(self.full_time_embedding_dim,
                         self.full_time_embedding_dim,
                         bias=False)

self.V = torch.nn.Linear(self.full_time_embedding_dim,
                         self.full_time_embedding_dim,
                         bias=False)
...
 query = self.W(time_embedding_query)
 key = self.V(time_embedding_key)
 scores = torch.matmul(query, key.transpose(-2, -1)) \
     / np.sqrt(query.size(-1))
```

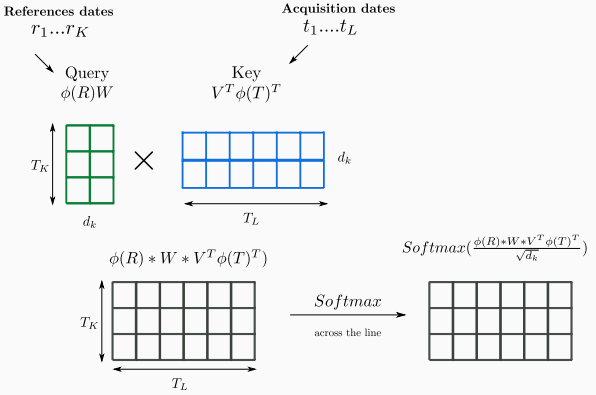


Figure 1: Attention score between reference time grid R and the acquisition T date

Time-driven attention score: masking attention sparse time series

Code

- (a) Replicate for all features

```
scores = scores.unsqueeze(-1).repeat_interleave(
    self.input_dimension,dim=-1)
```

- (b) Mask

```
masked_attention_scores = attention_scores.masked_fill(
    torch.logical_not(input_mask), -1e9)
```

- (c) Softmax feature or along acquisition dates

```
attention_scores_softmax = torch.nn.functional.softmax(
           masked_attention_scores, dim=-2)
```

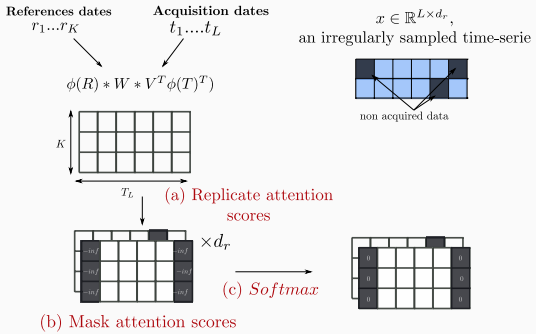


Figure 2: Masking the attention scores to handle sparse time series

Input signal interpolation

In the paper

$$\widehat{x_d}(r,t) = \Sigma_{i=1}^{Ld} k(r, t_{id}) x_{id}$$

In the code

(d) Multiplication with x + (e) concatenate

```
val_h = torch.sum(attention_scores_softmax*input_values,-2)
```

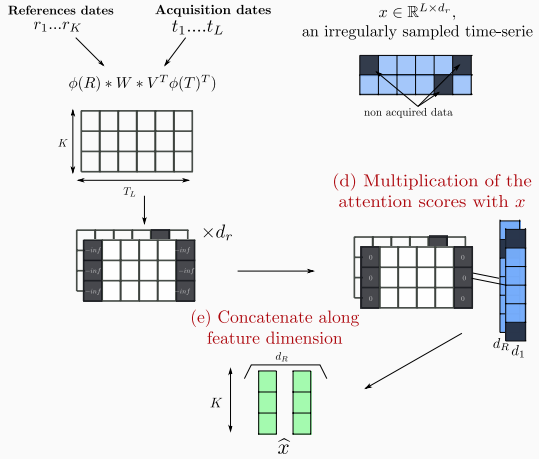


Figure 3: Interpolation of the input signal to the reference grid

The intensity term from unTAN

The intensity term measures the availability of dates required by attention with respect to dates in input signal

In the paper

We denote L_d the set of dates available for current signal (excluding masked dates) and L_u the set of all possible dates in the whole dataset

$$\alpha(r,t) = exp(\phi(r) w v^T \phi(t)^T \sqrt{d_K})$$

$$k_{L_d}(r,t) = \frac{\alpha(r,t)}{\Sigma_{t_i \in L_d} \alpha(r,t_i)}, k_{L_u}(r,t) = \frac{\alpha(r,t)}{\Sigma_{t_i \in L_u} \alpha(r,t_i)}$$

$$int(r,t) = \frac{k_{L_u}(r,t)}{k_{L_d}(r,t)} = \frac{\Sigma_{t_i \in L_d} \alpha(r,t_i)}{\Sigma_{t_i \in L_u} \alpha(r,t_i)}$$

In the code

```
all_attention_scores = self.attention_dot_product(reference_time_embedding, all_possible_input_times_embedding)
intensity = torch.logsumexp(masked_attention_scores, dim=-2)
intensity = intensity - torch.logsumexp(all_attention_scores,dim=-2)
intensity = torch.exp(intensity)
```

mTAN and unTAN wrap-up

Inputs

- A tensor of reference time points of shape [K]
- A tensor of acquisition times of shape [B,L]
- A tensor of input signals of shape [B,L,R]
- A tensor of input masks of shape [B,L,R], determining which channel is available at each of L dates

Outputs

A tensor of shape [B,H,K,(2x)R] containing for each attention head H:

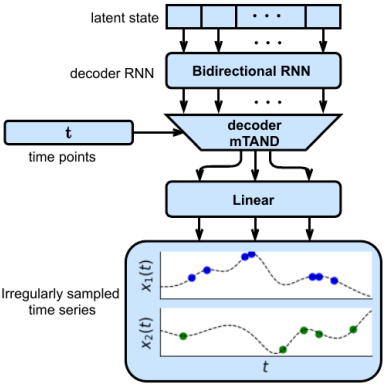
- the R input signals re-sampled at the K reference time points
- the R intensity term denoting the availability of attention required time points in input signal (optional)

Learned

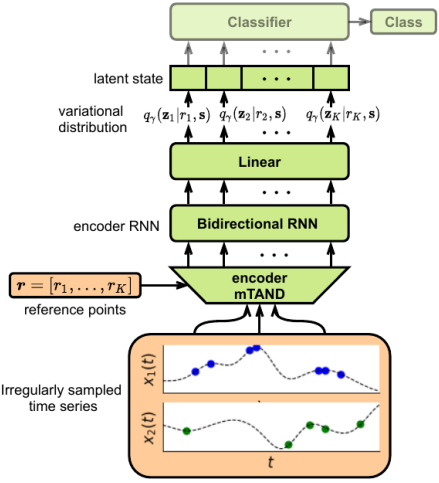
- The time-embedding linear layer
- V and W, the matrix that builds the softmax masked attention scores
- $\approx$ learned kernels for temporal interpolation

mTAN and unTAN Variational Auto-encoders architectures

mTAN VAE (figure from [1])

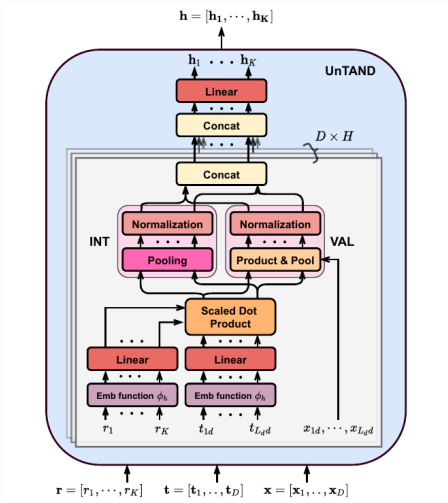


(a) Generative Model (Decoder)

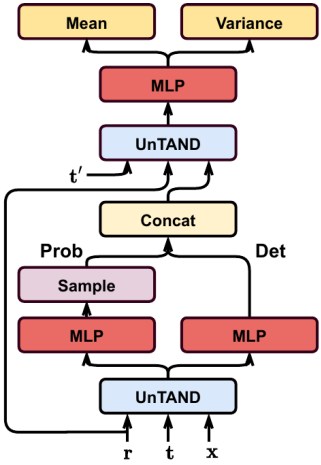


(b) Inference Network (Encoder)

HETVAE (figure from [2])



(a) Uncertainty Aware Multi-Time Attention Networks



(b) Heteroscedastic Temporal VAE

Main differences between both architectures

	mTanVAE	HETVAE
Time encoding module	mTAN	unTAN = mTAN + Intensity
Output variance	Fixed	Estimated
Loss function on output	Gaussian Negative Log-Likelihood	Gaussian Negative Log-Likelihood (+ a bit of MSE)
Loss function on latent	Normal Kullback-Liebler	Normal Kullback-Liebler
Time-aware encoder	Yes (Gated Recurrent Unit)	No (Plain MLP applied to each latent step)
Deterministic path	No	Yes

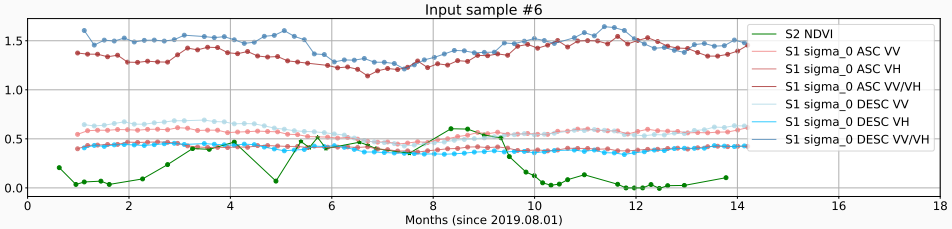
Some innovations of HETVAE can also be applied to mTAN VAE

- Intensity
- Estimation of output variance

Testing mTAN with SITS: experimental set-up

Dataset sampling from Pastis-R

[3] Garnot, V. S. F., Landrieu, L., & Chehata, N. (2022). Multi-modal temporal attention models for crop mapping from satellite time series. ISPRS Journal of Photogrammetry and Remote Sensing, 187, 294-305. (paper, code, data)



- Sample 24 330 Sentinel-2 and Sentinel-1 (asc and desc) time-series from Pastis-R patches
- Up to 10 samples per patches from all classes but background, averaged on 3x3 neighborhood
- MAJA-like cloud filtering rule (no cloud mask provided) for Sentinel-2, temporal averaging for Sentinel-1
- NDVI + random masking: random start, random length for each series

Experimental set-up (1/2)

Auto-encoding tasks

Monomodal task masked NDVI (1 input channel) ⇒ unmasked NDVI

Multimodal task masked NDVI + full Sentinel-1 (7 input channels) ⇒ unmasked NDVI

Models

	Task	Architecture	Use Intensity	Target variance	# params
`mono_mtanvae_woi_wovar`	Mono	mTAN VAE	No	0.01	45 351
`mono_mtanvae_woi_wvar`		mTAN VAE	No	Estimated	45 402
`mono_mtanvae_wi_wvar`		mTAN VAE	Yes	Estimated	45 498
`mono_hetvae_wi_wvar`		HET VAE	Yes	Estimated	44 020
`multi_mtanvae_wi_wvar`	Multi	mTAN VAE	Yes	Estimated	46 650
`multi_hetvae_wi_wvar`		HET VAE	Yes	Estimated	46,420

Latent space dimension 8

Time embedding per head 64

Number heads 2 for encoder, 1 for decoder

Reference time points 1 point every 5 days over the dataset date range

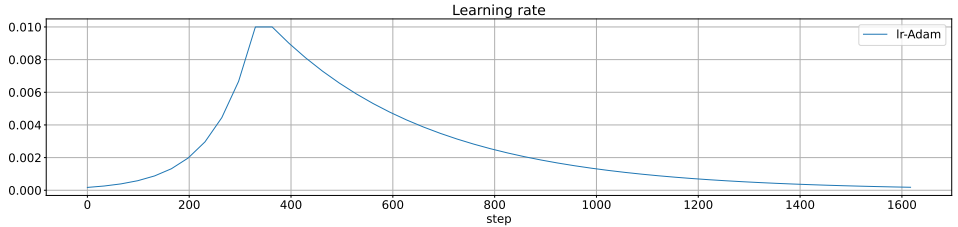
Experimental set-up (2/2)

Dataset splits

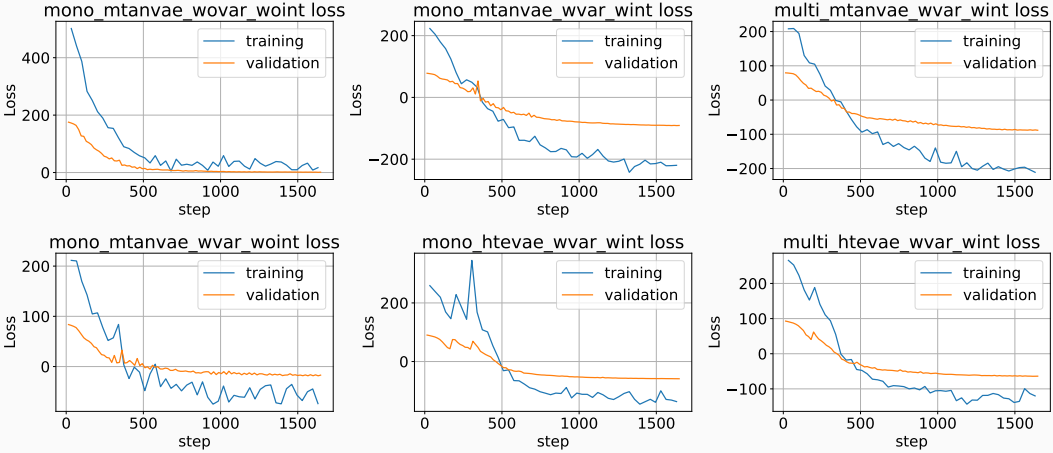
- Separate test set of 4 886 series never seen during training (except for mean and std estimation . . .)
- Among the 19 464 remaining series : randomly keep 2 433 (10% of total) for validation and the rest for training

Training

- Standard Adam Optimizer with learning rate 0.01, batch size 200, and 50 epochs
- Learning rate warm-up for 10 epochs and gentle decrease from 10 to 50 epochs
- All training occurs with standardized input / output data. Metrics are computed on unstandardized output

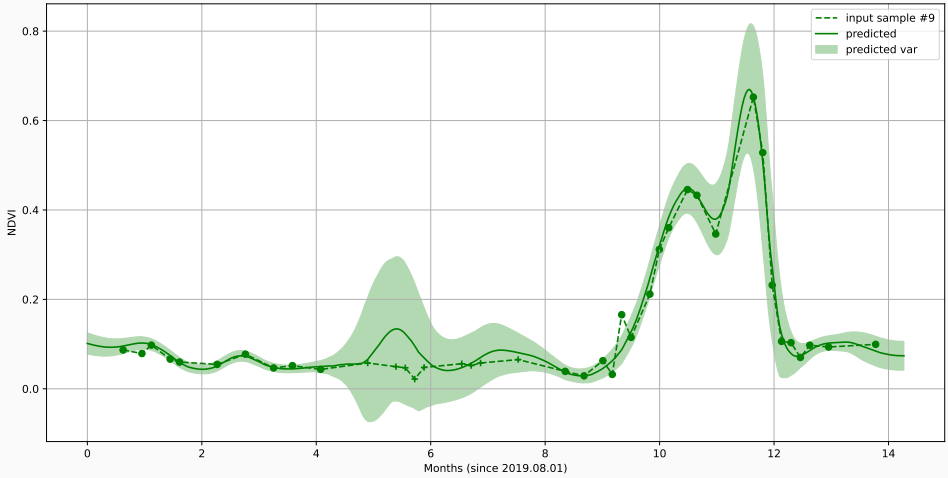


Evolution of training and validation losses during training

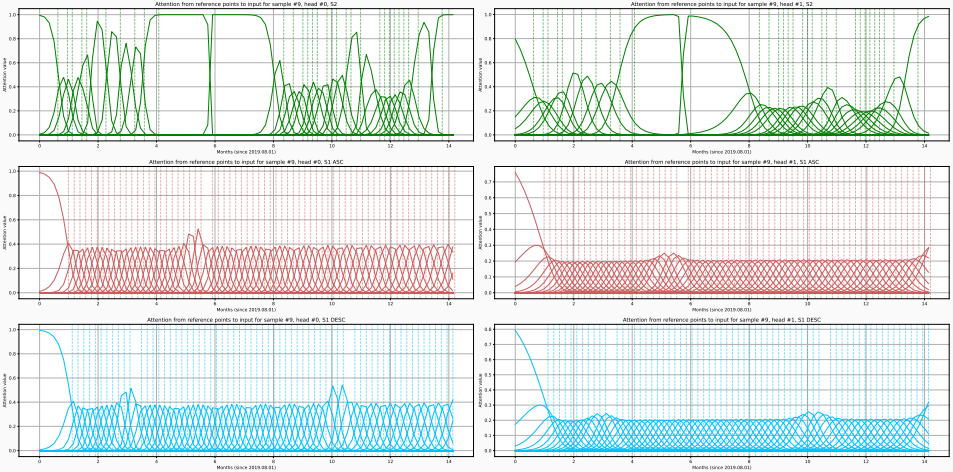


Anatomy of mTAN Variational Auto-Encoder for SITS

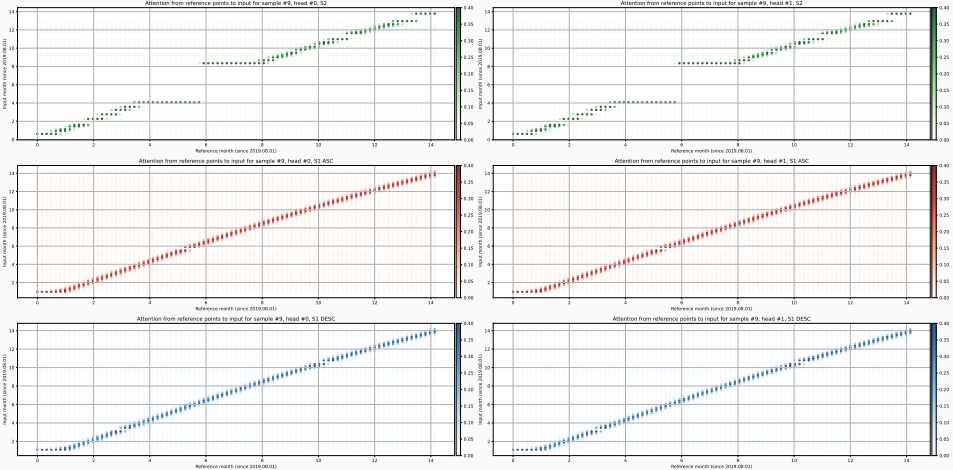
Anatomy of `multi_mtanvae_wi_wvar`: predictions



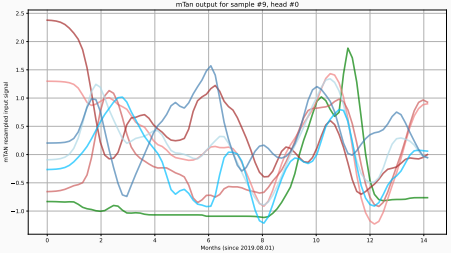
Anatomy of `multi_mtanvae_wi_wvar`: mTAN encoder attention

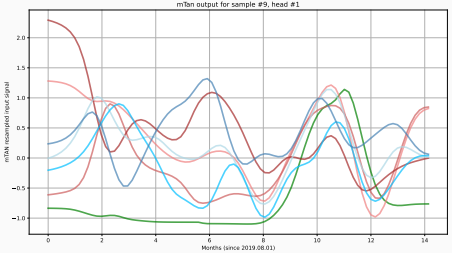


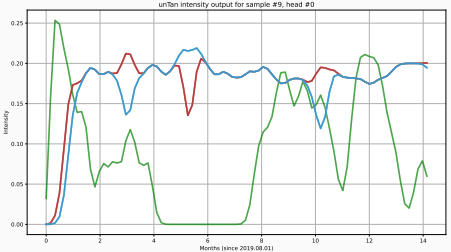
Anatomy of `multi_mtanvae_wi_wvar`: mTAN encoder attention

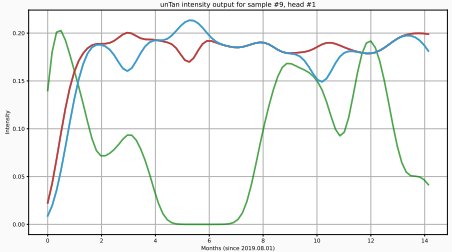


Anatomy of `multi_mtanvae_wi_wvar`: mTAN encoder output

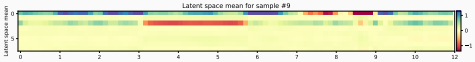


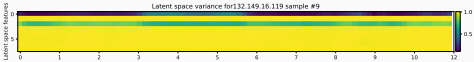


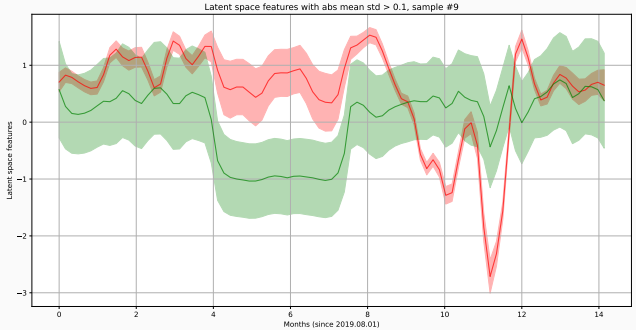




Anatomy of `multi_mtanvae_wi_wvar`: latent space features







Performance analysis

Quantitative analysis on separate testing set

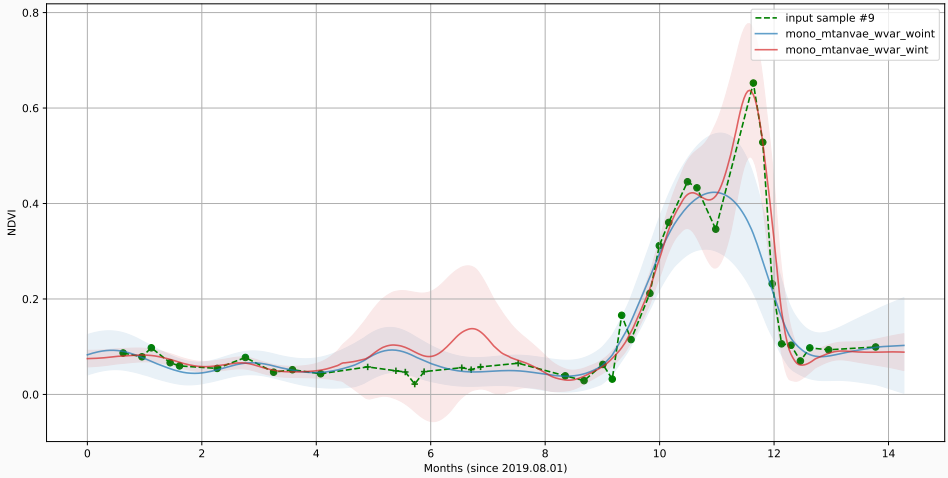
All target dates (masked and unmasked input dates)

	MSE	MAE	MAPE (%)	CIBR 1 σ (%)	MCIW
mono_mtanvae_wovar_woint	0.0990707	0.0624423	33.6582	82.4121	0.1
mono_mtanvae_wvar_woint	0.103249	0.0607378	38.1664	80.2746	0.0878534
mono_mtanvae_wvar_wint	0.0888043	0.0504897	53.1954	75.5675	0.0652271
mono_htevae_wvar_wint	0.101161	0.0616576	36.0062	78.3032	0.0902833
multi_mtanvae_wvar_wint	0.0890551	0.0515513	31.9344	70.9145	0.0576495
multi_htevae_wvar_wint	0.0955975	0.0575327	48.7533	78.9364	0.083374

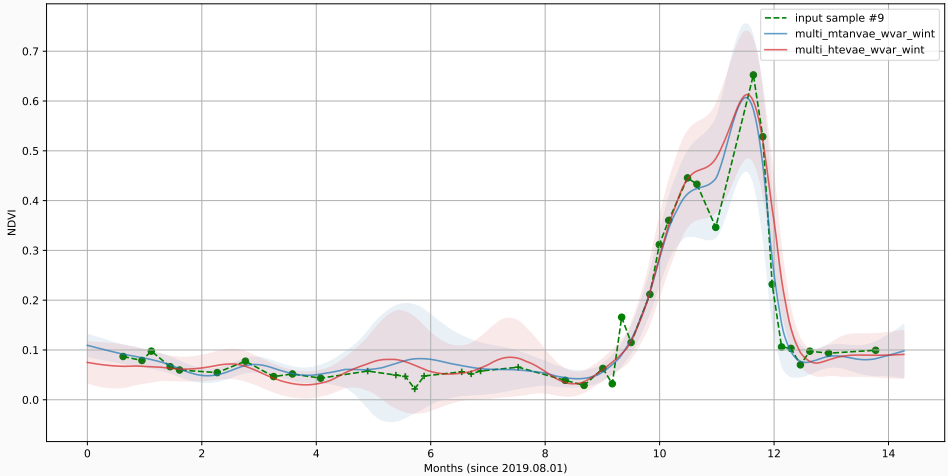
Only masked input dates

	MSE	MAE (%)	MAPE	CIBR σ (%)	MCIW
mono_mtanvae_wovar_woint	0.163771	0.114091	53.8516	60.2789	0.1
mono_mtanvae_wvar_woint	0.171507	0.117842	69.6694	55.4546	0.0965285
mono_mtanvae_wvar_wint	0.159295	0.116477	55.4961	67.5348	0.133646
mono_htevae_wvar_wint	0.162305	0.114302	52.4242	74.4151	0.150324
multi_mtanvae_wvar_wint	0.149947	0.10647	46.771	50.0518	0.0850109
multi_htevae_wvar_wint	0.159601	0.111383	49.8329	73.7114	0.142638

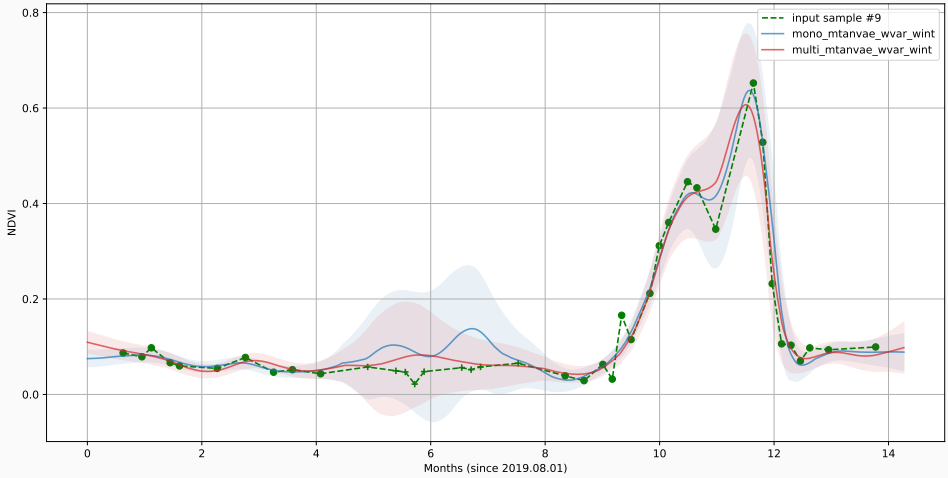
Intensity term helps predicting meaningful variance



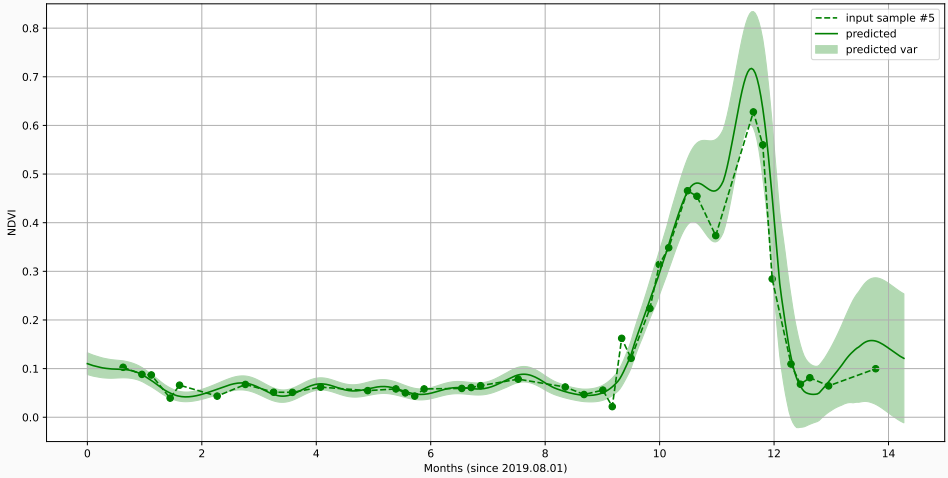
mTAN VAE with intensity and estimated variance outperforms HETVAE



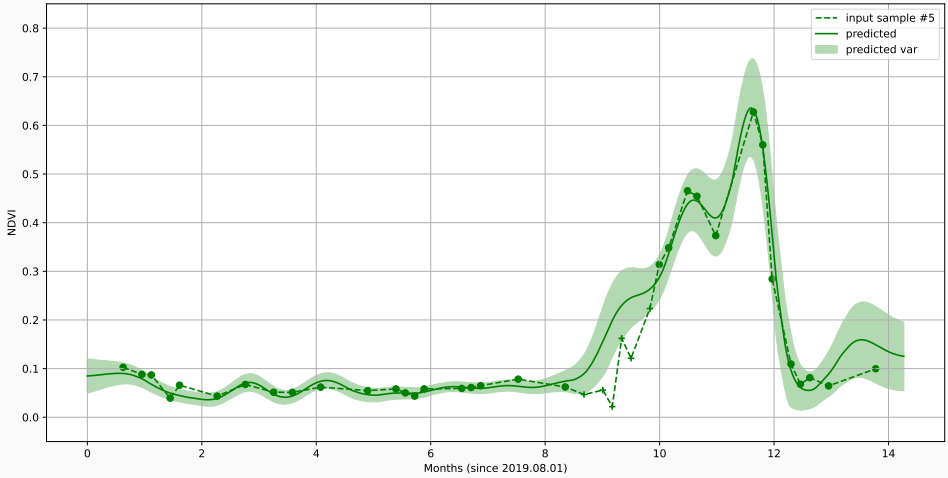
Multi-modal VAE seems to predict narrower variance



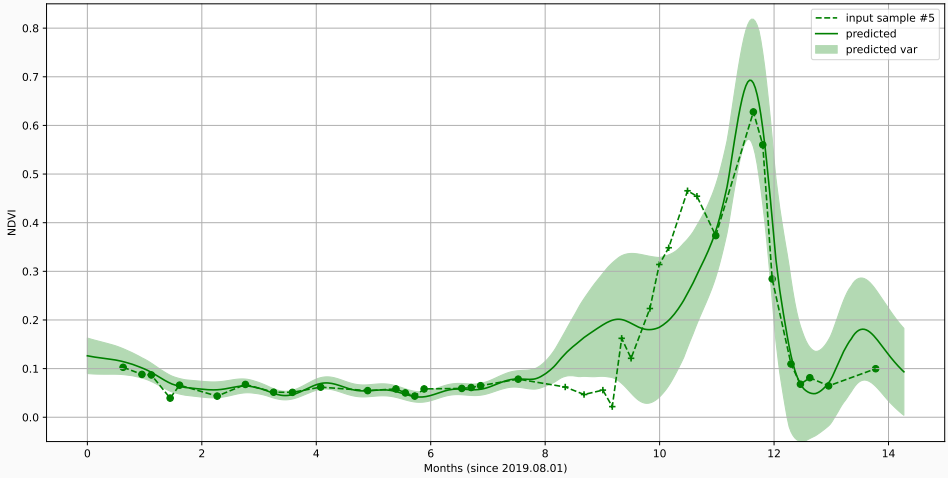
Behaviour of multi-modal mTAN VAE wrt. gradually larger optical cloud gaps



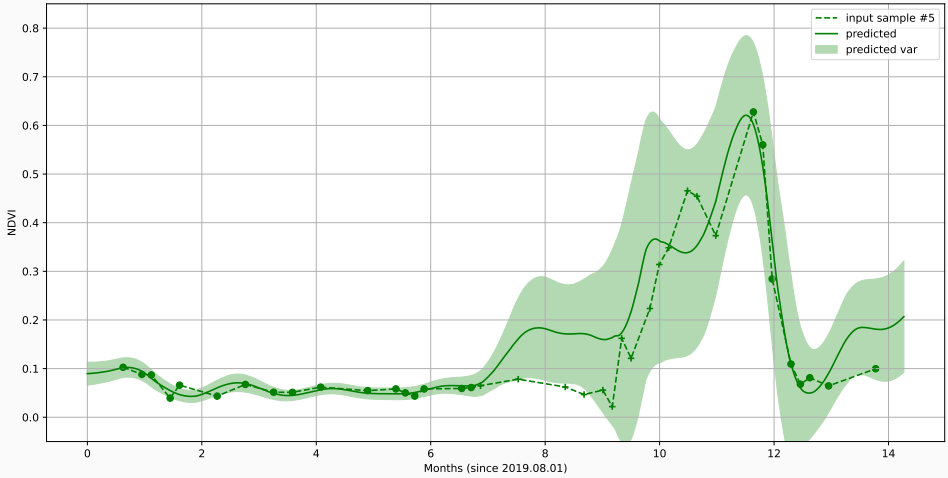
Behaviour of multi-modal mTAN VAE wrt. gradually larger optical cloud gaps



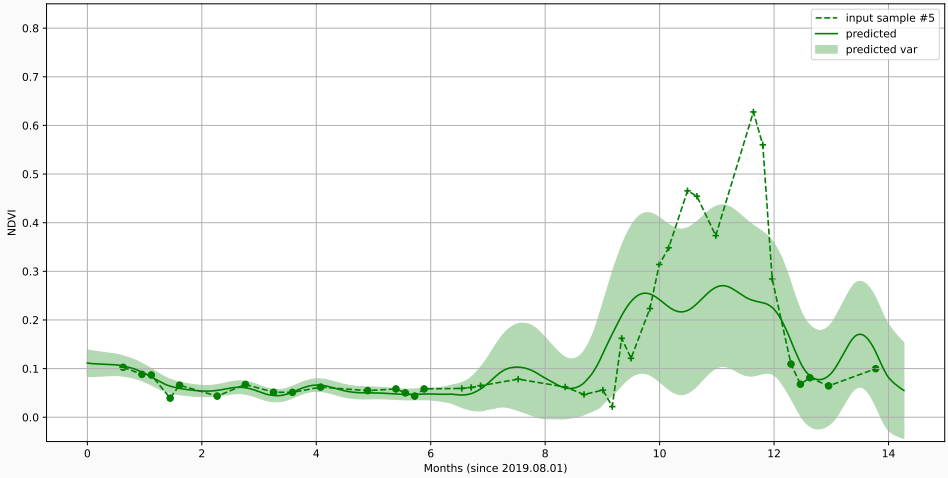
Behaviour of multi-modal mTAN VAE wrt. gradually larger optical cloud gaps



Behaviour of multi-modal mTAN VAE wrt. gradually larger optical cloud gaps



Behaviour of multi-modal mTAN VAE wrt. gradually larger optical cloud gaps



Conclusions

Conclusions

mTAN and unTAN

- mTAN [1] re-sample the input signals on a set of reference dates with learnable kernels
- It can handle irregular sampling and different sampling for multi-modal signals
- Intensity term from unTAN [2] measures the availability of dates required by attention mechanism in input signal

Variational Auto-Encoders based on mTAN and unTAN

- mTAN [1] and unTAN (mTAN + intensity) [2] can be used in Variational Auto-Encoders architectures
- Main innovations from [2] (intensity and estimation of output variance) can be added to mTAN VAE, which then outperforms HETVAE proposed in [2] (but the latter has less params)
- Intensity term helps predicting meaningful output variance, related to missing data

Application to multi-modal estimation of gap-free NDVI time-series

- A proof of concept, for the sake of the demonstration! Many things missing for a valid assessment
- Shows that mTAN derived VAEs provide interesting results for SITS with interpretable by-products
- Interest of VAEs is latent variational space, not output! Needs to be assessed on other tasks (for instance classification)

Perspectives

Those papers helped us revisit our comprehension of attention

- Transformers mix self attention and positional encoding, which makes it hard to see but ...
- Attention is fundamentally a learnable kernel smoothing mechanism!

Tsai, Y. H., Bai, S., Yamada, M., Morency, L., & Salakhutdinov, R. (2019). Transformer dissection: a unified understanding of transformer's attention via the lens of kernel. CoRR (paper found by Jordi)

mTAN and unTAN are useful pieces for any regression/classification problem implying SITS

- Alternative to gap-filling and re-sampling that can be trained end-to-end and is still fairly interpretable
- Need to be combined with other modules for a full scheme:
  - Self attention for input signal driven interest, for instance using a down-stream transformer
  - CNN-based architecture for spatial context encoding (for instance using an up-stream U-Net-like)

Open questions and limitations

- Static reference time points

Code and Licence

Code available on `https://gitlab.cesbio.omp.eu/activites-ia/torchmuntan`

(requires CESBIO gitlab account)