Semi-Supervised Learning for Spatio-Temporal Segmentation of Satellite Images

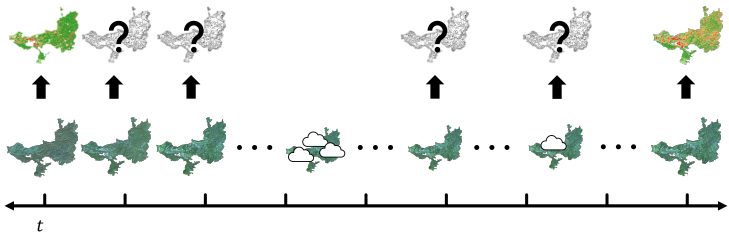
Antonín Hruška

June 13, 2023

Motivation

ESA project: Enhanced Spatiotemporal Land Change Monitoring Based on Sentinel-2 Time Series and VHR Images

Main motivation: Segment time series of satellite imagery.



Challanges:

- Only 2% of the data is annotated. The rest is unlabeled. (5/250)
- Partial occlusion in the data (clouds, snow) = missing measurements
- No validation data

Proposed solutions:

- Apply Semi supervised learning and generative models
- Evaluate the algorithms on the CityScape dataset instead of satellite images.

Considered SSL methods

Semi Supervised learning:

$$\mathcal{D} = \mathcal{X} \cup \mathcal{U} : \mathcal{X} = \{(\mathrm{x}_1, y_1), \ldots, (\mathrm{x}_l, y_l)\}, \quad \mathcal{U} = \{(\mathrm{x}_{l+1}), \ldots, (\mathrm{x}_u)\},$$

where x_i are features and y_i are labels.

1. **Discriminative model** + MixMatch
2. **Generative model** (hiearchical variational autoencoder) + symmetric training

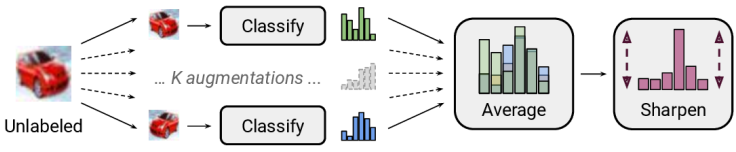
Main contributions: Adapting both methods to the task of semantic segmentation. Implementation and comparison of both methods on CityScape dataset.

MixMatch

Holistic method combining the data augmentation, pseudo-labeling, entropy minimization and Mix Up procedure:

$$\mathcal{X}', \mathcal{U}' = \text{MixMatch}(\mathcal{X}, \mathcal{U}, T, K, \alpha)$$

$$\mathcal{L} = \frac{1}{|\mathcal{X}'|} \sum_{x', p' \in \mathcal{X}'} H(p', f_\theta(x')) + \lambda_\mathcal{U} \frac{1}{L|\mathcal{U}'|} \sum_{u', q' \in \mathcal{U}'} ||q' - f_\theta(u')||_2^2$$



(a) Pseudo labeling

(b) Mix Up

Figure: MixMatch procedure. *MixMatch: A Holistic Approach to Semi-Supervised Learning*

Symmetric learning for hiearchical variational autoencoders

Generative latent variable **model** with *decoder*

$$p_\theta(x, z) = p_\theta(z)p_\theta(x \mid z)$$

and the *inference model* (*encoder*):

$$q_\phi(z|x) \approx p_\theta(z|x)$$

Standard evidence lower bound (ELBO) objective

$$\mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x, z)}{q_\phi(z|x)}\right] = \mathbb{E}_{q_\phi(z|x)} \log p_\theta(x) - \mathrm{KL}(q_\phi(z|x)||p_\theta(z|x)) \to \max_{\theta,\phi}$$

is not applicable if latent z contains segmentations. Symmetric equilibrium training separtes the log-likelihood of both encoder and decoder:

$$\mathcal{L}_p(\theta, \phi) = \mathbb{E}_{\pi(x,z)}\left[\log p_\theta(x, z)\right] + \mathbb{E}_{\pi(z)}\left[\log p_\theta(z)\right] + \mathbb{E}_{\pi(x)}\,\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x, z)\right] \to \max_\theta$$

$$\mathcal{L}_q(\theta, \phi) = \mathbb{E}_{\pi(x,z)}\left[\log q_\phi(z|x)\right] + \mathbb{E}_{\pi(z)}\,\mathbb{E}_{p_\theta(x|z)}\left[\log q_\phi(z|x)\right] \to \max_\phi$$

The learning corresponds to Nash-equilibrium 2-player game with the above objectives.

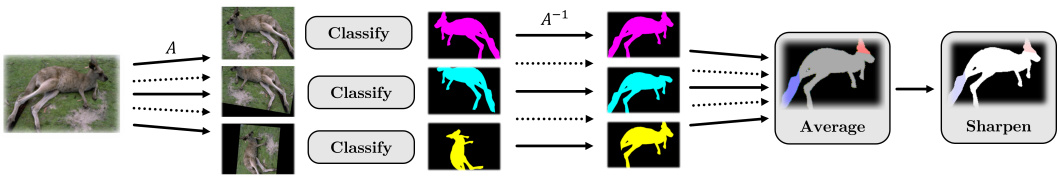
Mixmatch adaptation and methods

Experiments settings:

- We use the CityScape dataset to evaluate the methods for semantic segmentation.
- We use plain accuracy and IoU as metrics.
- Both discriminative model and the encoder of HVAE have U-net shape.

MixMatch adaptation

- The pseudolabeling procedure requires to apply the augmentation on the segmentation. The augumentation has to be invertible in order to compute average.
- We propose to use the affine transformations, which are rich and also invertible.

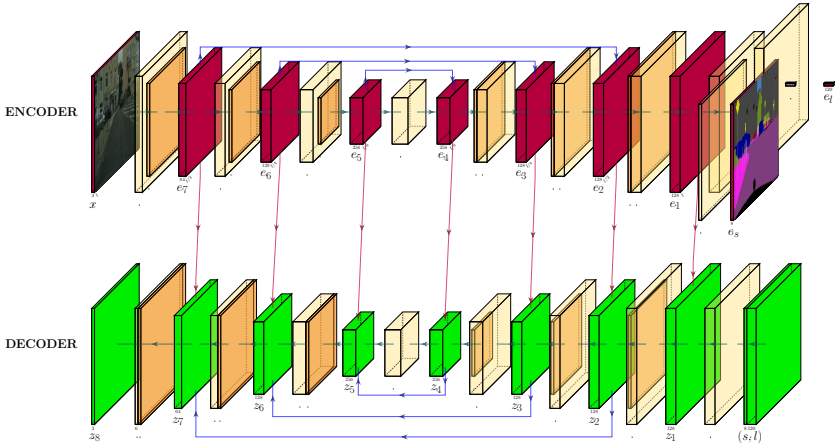


Hiearchical variational autoencoder adaptation

- binary hiearchical VAE

$$z = (z_0, z_1, \ldots, z_8), \quad z_0 = (s, l),\ z_8 = x$$

where s is categorical distribution to model segmentation, and l is latent code.



Mixmatch results (tables)

Images [#]	10 (0.33%)	100 (3.3%)	500 (16.8%)	1000 (33.6%)	2975 (100%)
Mixmatch	84.53	90.76	93.42	94.32	94.84
Supervised	76.85	87.59	93.50	94.71	95.58

Table: Mixmatch accuracy rate (%) on CityScape dataset. The first row contains the number of fully annotated images avilable during the training. In the last column, the CityScape dataset is replicated and used as labeled and unlabeled dataset.

Images [#]	10 (0.33%)	100 (3.3%)	500 (16.8%)	1000 (33.6%)	2975 (100%)
Mixmatch	48.10	61.98	69.24	71.97	73.08
Supervised	41.78	54.38	68.25	71.93	73.84

Table: Mixmatch average IoU (%) on CityScape dataset

Mixmatch results (images)

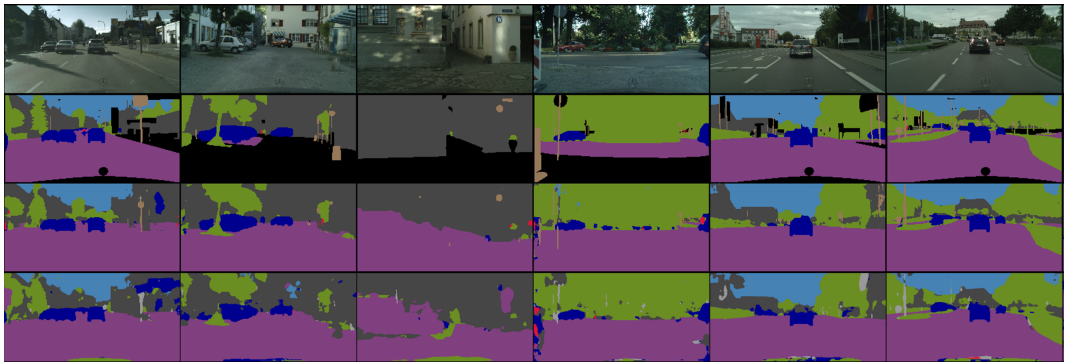


Figure: Images of models, which where trained on **100** annotated images. The first two rows contain the image and its ground truth segmentation. The MixMatch predictions are shown in the third row, while the supervised baseline predictions are in the fourth.

Symmetric equilibrium training results

Images [#]	10 (0.33%)	100 (3.3%)	500 (16.8%)	1000 (33.6%)	2975 (100%)
HVAE	74.01	86.18	90.23	92.63	93.51
Supervised	74.82	87.73	90.81	94.18	94.93

Table: HVAE accuracy rate (%)

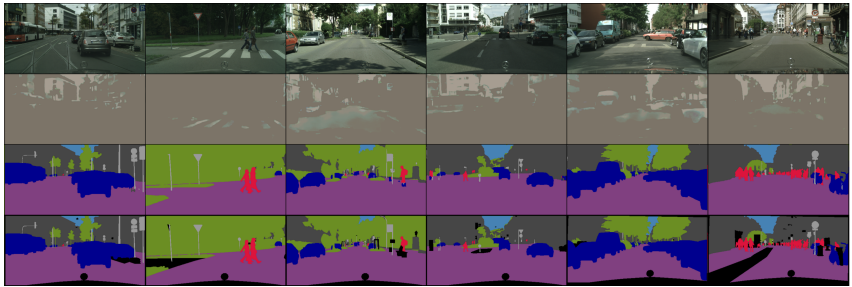


Figure: The first two rows are filled with the original and reconstructed images. The third and fourth rows contain model and ground truth segmentation.

Conclusion

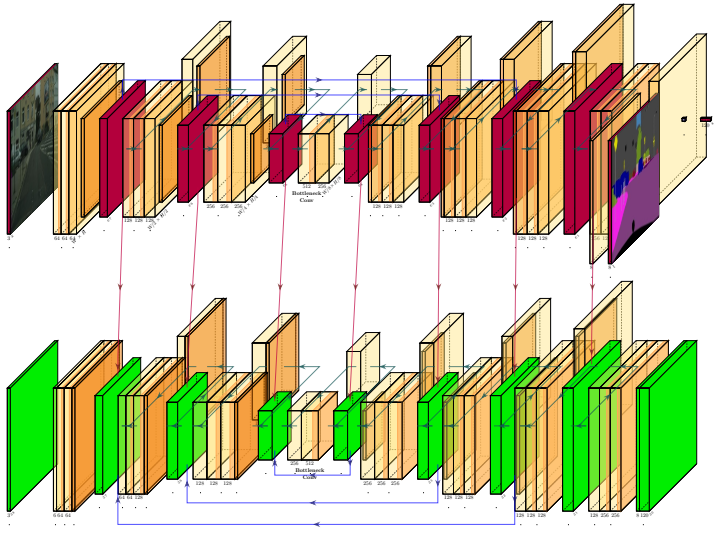
- The mixmatch can improve the results especially in scenario with low amount of annotated data.
- The HVAE in its current state is not capable of generating the images, which could improve the encoder accuracy.
- Despite its limitations, the HVAE shows promise as a potential approach worth exploring further.

Reviewer's questions (1/2)

Unfortunately, the final architecture used for symmetric learning is not explained to the necessary level of detail. Basically, it is only said that both encoder and decoder have a U-Net like architecture. Note however that they share parameters, i.e., the encoder uses decoder parts in order to e.g., generate from $q_{\theta,\phi}(z \mid x)$. It is not entirely clear how it works. Are the stochastic variables z_i attached to all resolution levels? Are they attached to both U-Net branches (of decreasing and increasing spatial resolutions)? Is there a skip-connection also between blocks of the original resolution? How many resolution levels there are? So, a more detailed explanation as well as a figure illustrating the architecture and generating / learning process would be highly appreciated.

An important question is the dimensionality of the latent z-s, especially for l (which is a part of z_0). Note that the segmentation alone does not include any coloring information, like segment colors (or colors of objects / instances), textures, shadows etc. Hence, in order to generate realistically looking images, such information should be encoded by the latent variables, in particular by l. If its dimension is low, it is obviously not capable to represent this information adequately (btw. if so, perhaps it can explain why the reconstructed images are bad). If, however the dimension of l is high, one has a gigantic input tensor for the decoder (due to replication of l along spatial dimensions), which obviously causes certain technical problems.

Detailed architecture



Reviewer's questions (2/2)

Concerning experiments. It is somewhat surprising that there are no "baseline" experiments for symmetric learning like it was done for Mixmatch. Moreover, such baseline experiments could be designed in different ways. For example, one could just learn the segmentation model $q_\theta(s|x)$ on fully supervised training data of different sizes. Next, one can apply the "full" symmetric learning according to eqs. (1), (2) but again on fully supervised training data of different sizes only. Hopefully, comparing these two experiments one can observe some improvement, because additional terms in eqs. (1), (2) should serve as a regularizer for the segmentation model, and hence improve generalization capabilities. Finally, comparing the final experiments (which are present in the work) with fully supervised symmetric learning, one can draw conclusions about the applicability of the symmetric learning for SSL.

$$\mathcal{L}_p(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{\pi(\mathsf{x},\mathsf{s})}\, \mathbb{E}_{q_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathsf{z}_{>0},\mathsf{l}|\mathsf{x},\mathsf{s})} \big[\log p_{\boldsymbol{\theta}}(\mathsf{x},\mathsf{z})\big] + \mathbb{E}_{\pi(\mathsf{x})}\, \mathbb{E}_{q_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathsf{z}|\mathsf{x})} \big[\log p_{\boldsymbol{\theta}}(\mathsf{x},\mathsf{z})\big] \tag{1}$$

$$\mathcal{L}_q(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{\pi(\mathsf{x},\mathsf{s})} \big[\log q_{\boldsymbol{\phi}}(\mathsf{s} \mid \mathsf{x})\big] + \mathbb{E}_{\pi(\mathsf{s})}\, \mathbb{E}_{p(\mathsf{l})}\, \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathsf{x},\mathsf{z}_{>0}|\mathsf{z}_0)} \big[\log q_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathsf{z}|\mathsf{x})\big] \tag{2}$$