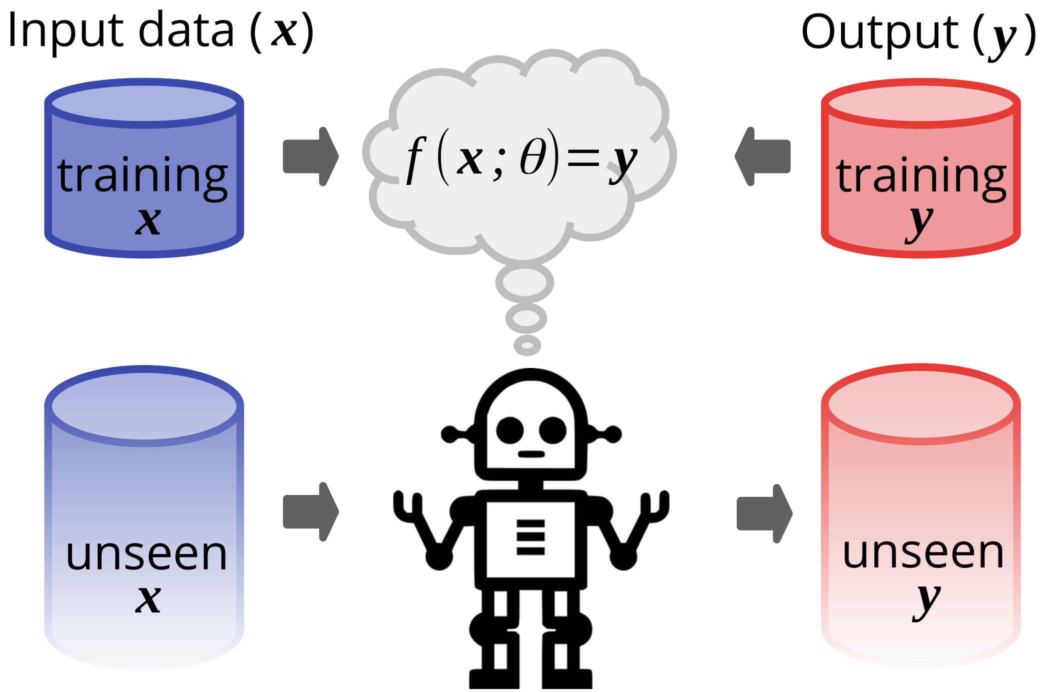
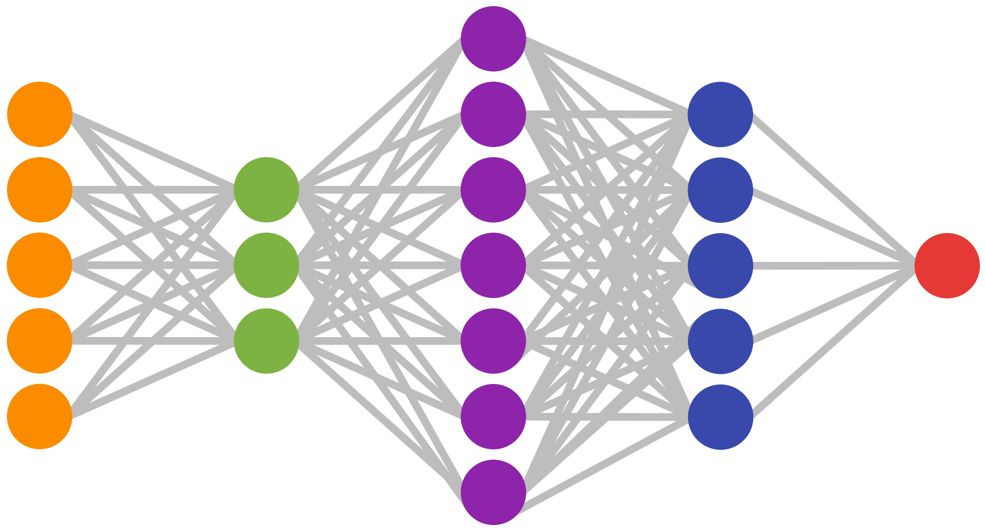
Label-efficient Deep Learning in Remote Sensing

Michael Mommert, University of St. Gallen

Resources: github.com/mommermi/iadfschool2023_efficientlearning

Introduction



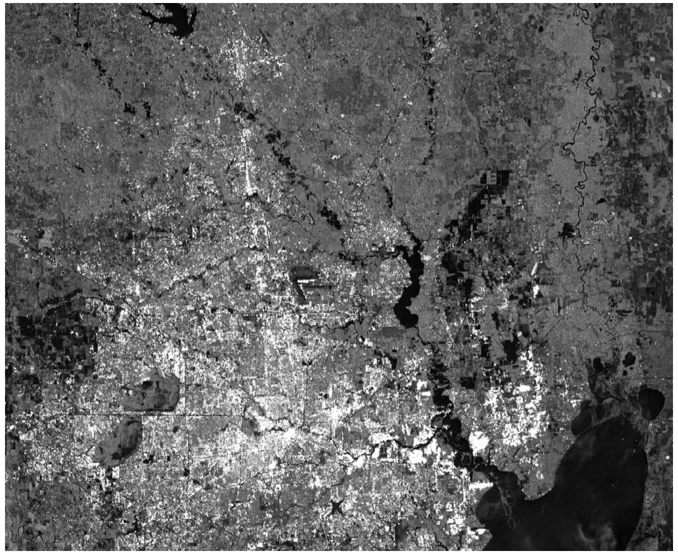


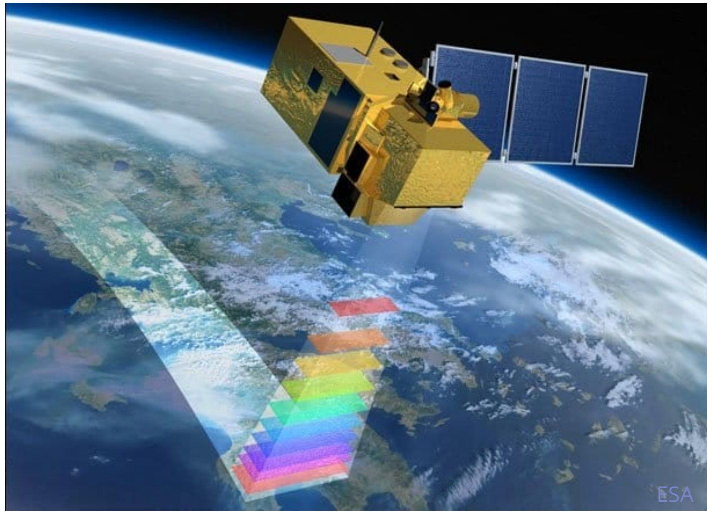
Deep Learning for Earth observation

Earth observation data are highly complex (unstructured, multi-modal).

How can we analyze these vast amounts of data?

Deep Learning offers the **scalability** to analyze large amounts of data.





Deep Learning for Earth observation

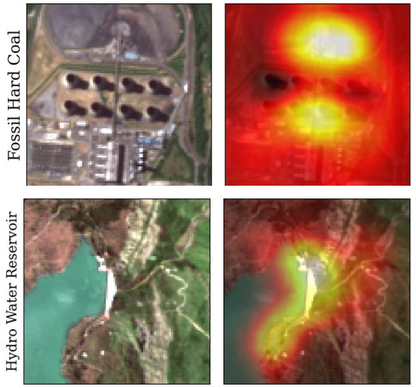
Earth observation data are highly complex (unstructured, multi-modal).

How can we analyze these vast amounts of data?

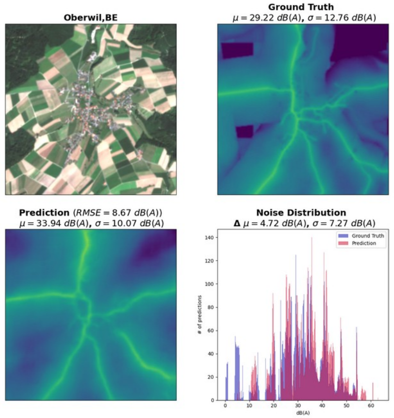
Deep Learning offers the **scalability** to analyze large amounts of data.

Deep Learning also offers the **flexibility** to deal with a range of different tasks.

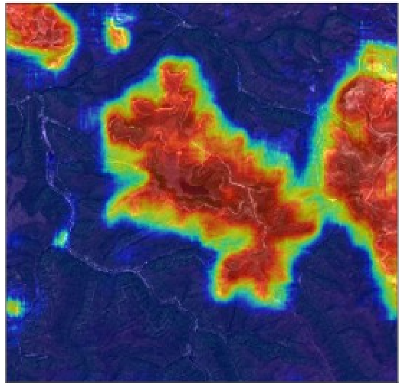
How does it work?



Classification



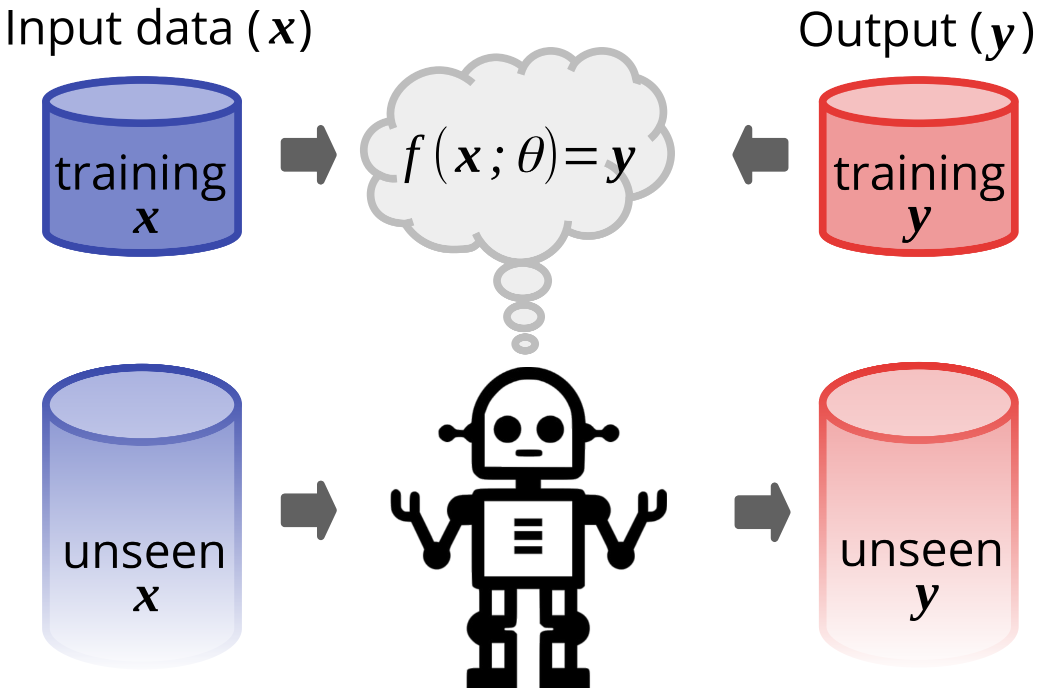
Regression

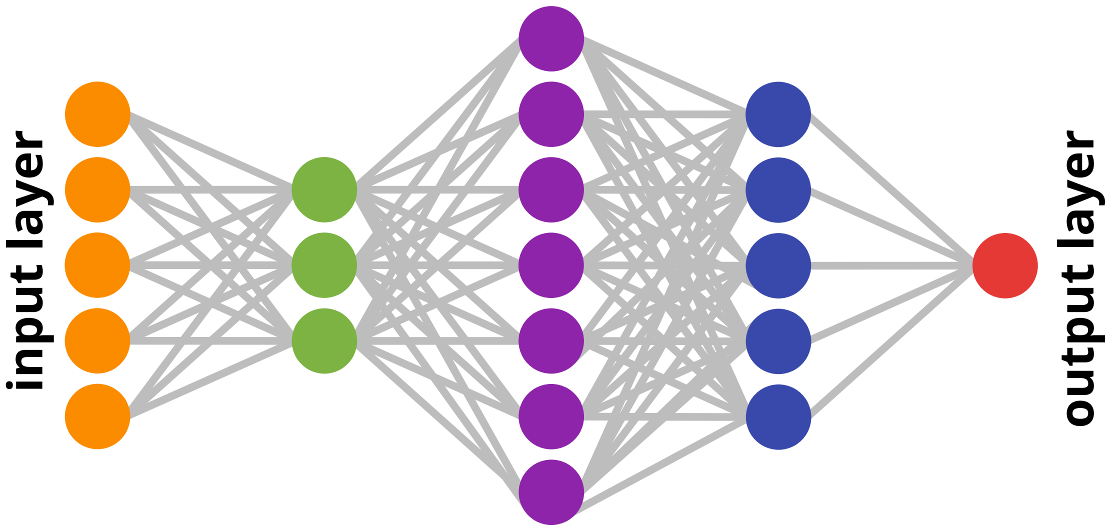


Segmentation

Object Detection

Supervised learning with Neural Networks





A machine learns a task from **annotated examples**.

Mathematically, it learns a function, f, that maps input data, $\boldsymbol{x}$, to the output, $\boldsymbol{y}$.

A Neural Network is a cascade of mathematical functions; each neuron contains learnable weights that represent the learned knowledge.

How does the model learn?

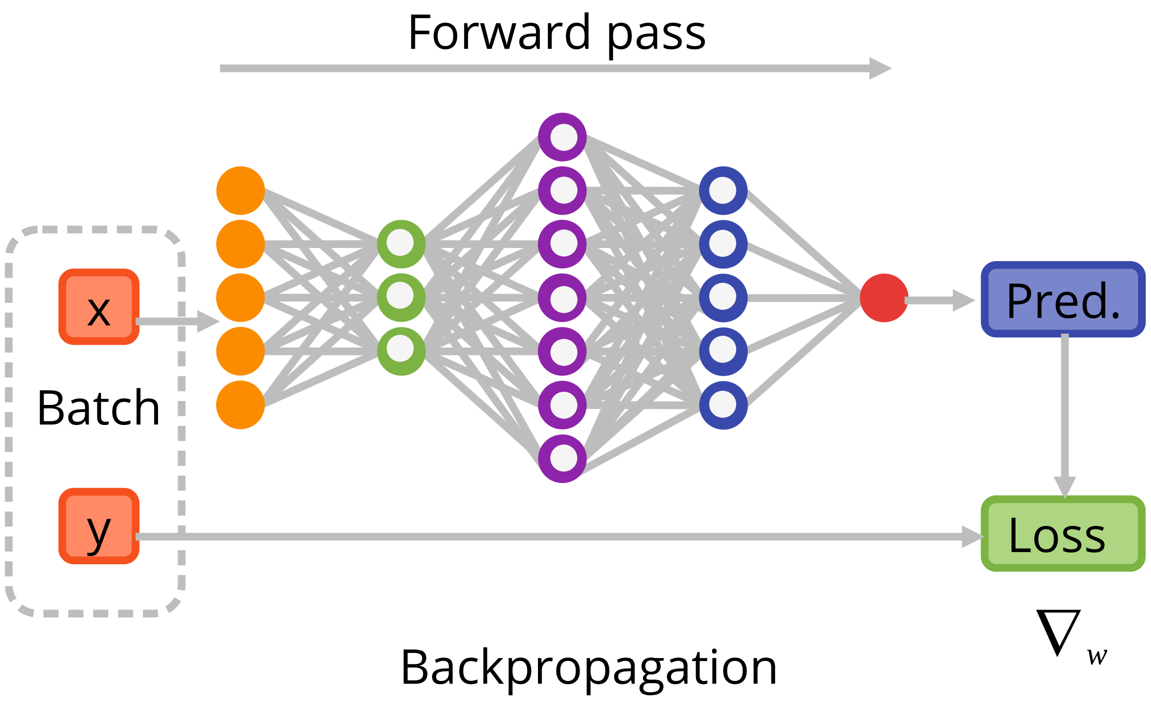
Neural network training pipeline

- Sample batch (input data x and target data y) from training dataset:

1 epoch

  - Evaluate model on batch input data (=prediction) in forward pass
  - Compute loss on prediction and target y
  - Compute weight gradients with backprop.
  - Modify weights based on gradients and learning rate
  - Repeat for all batches

- Repeat for a number of epochs, monitor training and validation loss + metrics
- Stop before overfitting sets in

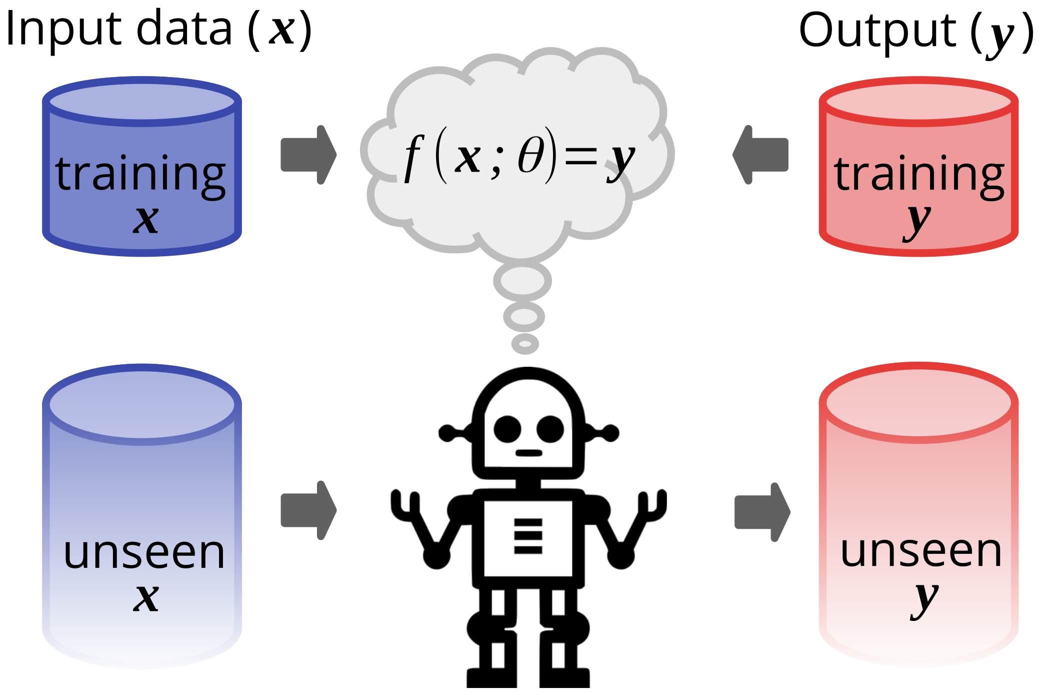


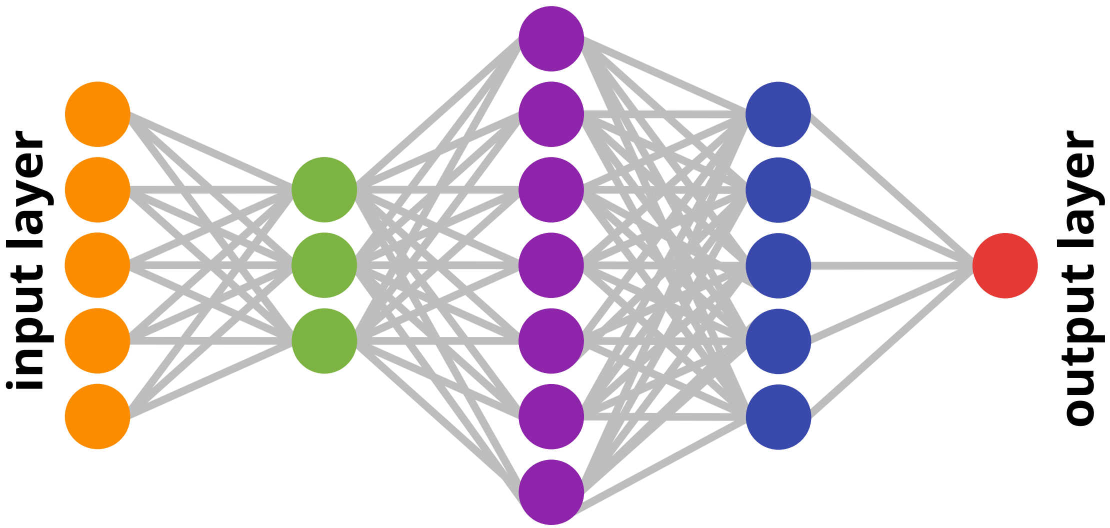
Let’s implement a fully supervised learning pipeline with PyTorch and PyTorch Lightning!

Please go to:

github.com/mommermi/iadfschool2023_efficientlearning

Supervised learning with Neural Networks



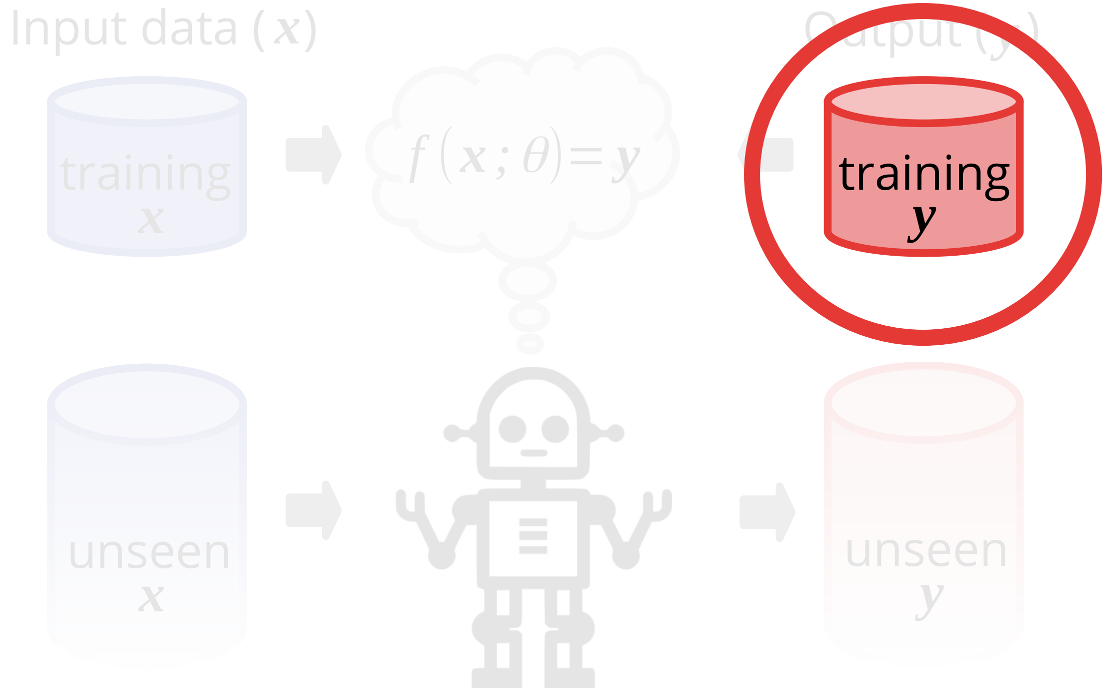


A machine learns a task from **annotated examples**.

Mathematically, it learns a function, f, that maps input data, $\boldsymbol{x}$, to the output, $\boldsymbol{y}$.

A Neural Network is a cascade of mathematical functions; each neuron contains learnable weights that represent the learned knowledge.

Supervised learning with Neural Networks



The availability of annotations typically represents the most important **bottleneck** in supervised learning.

Can we force the model to use the available annotations more **efficiently**?

Can we take advantage of the vast amounts of **unannotated data**?

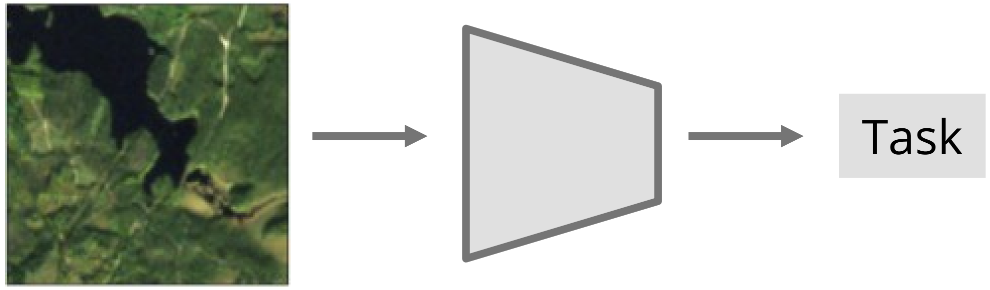
A machine learns a task from **annotated examples**.

Mathematically, it learns a function, f, that maps input data, x, to the output, y.

A Neural Network is a cascade of mathematical functions; each neuron contains learnable weights that represent the learned knowledge.

How can we use annotated data more efficiently?

- Data augmentations



How can we use annotated data more efficiently?

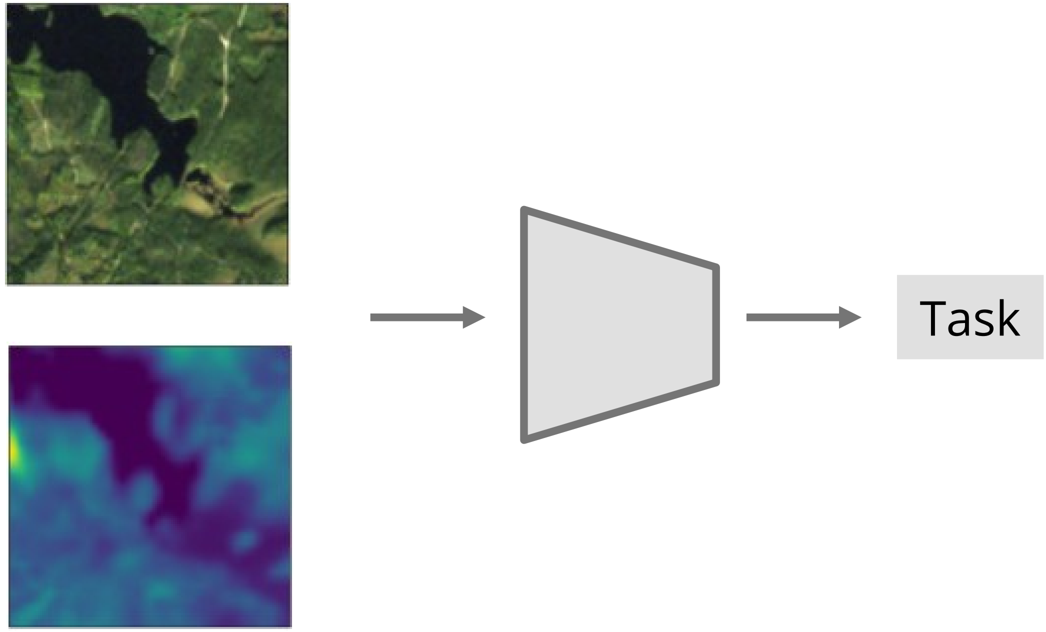
- Data augmentations

How can we use annotated data more efficiently?

- Data augmentations
- Data Fusion

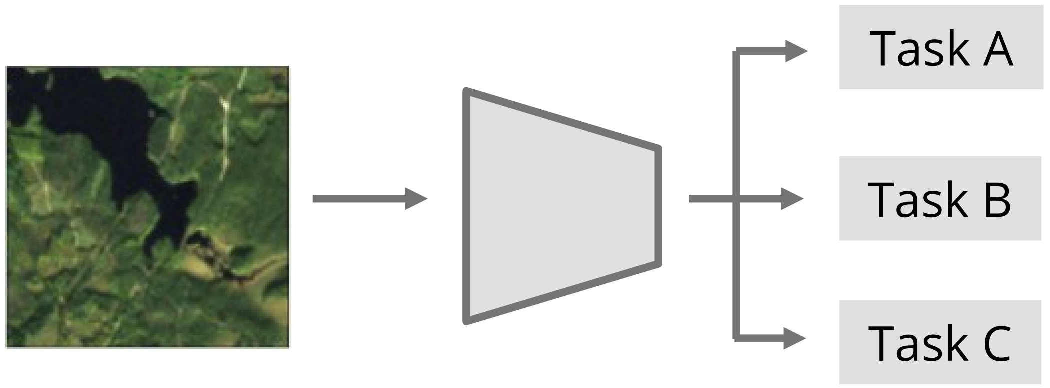
multispectral

other modality
(DEM)



How can we use annotated data more efficiently?

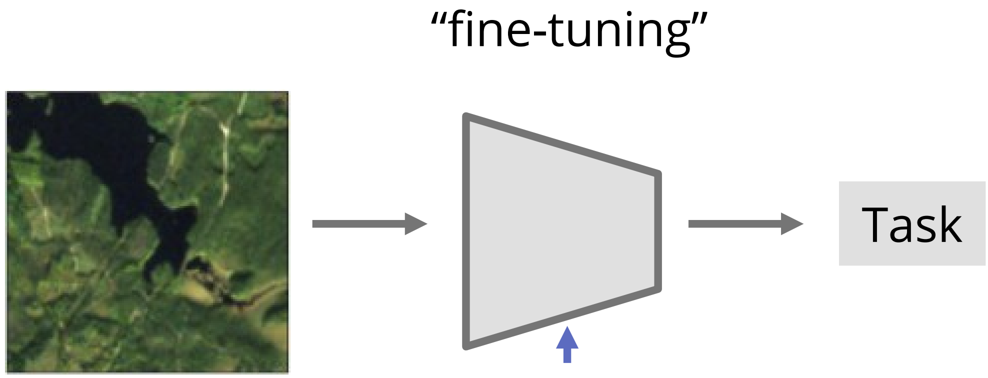
- Data augmentations
- Data Fusion
- Multi-task Learning

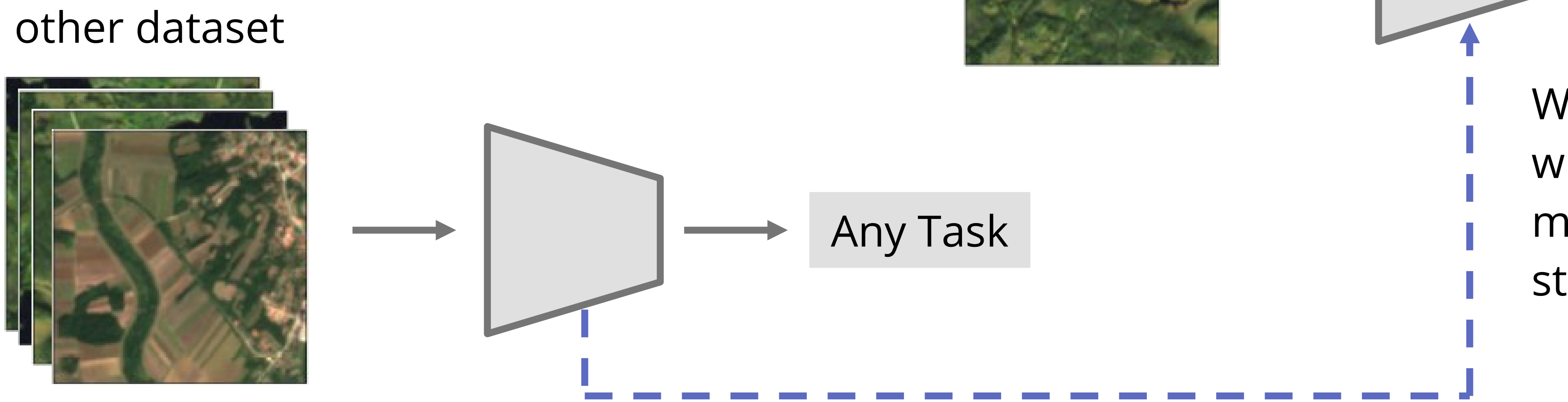


How can we use annotated data more efficiently?

- Data augmentations
- Data Fusion
- Multi-task Learning
- Transfer Learning

Can we pretrain a model from unannotated data?

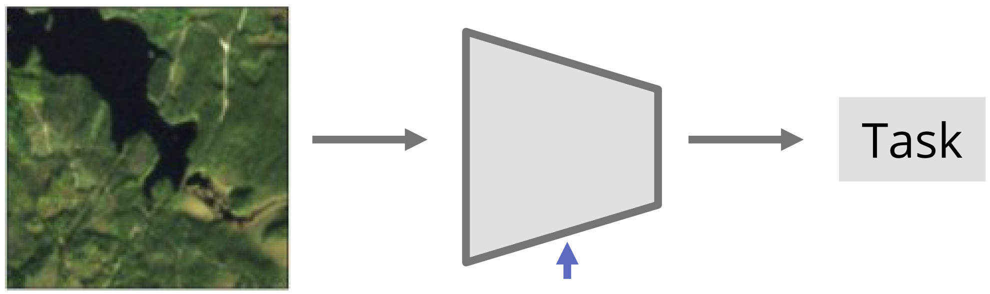


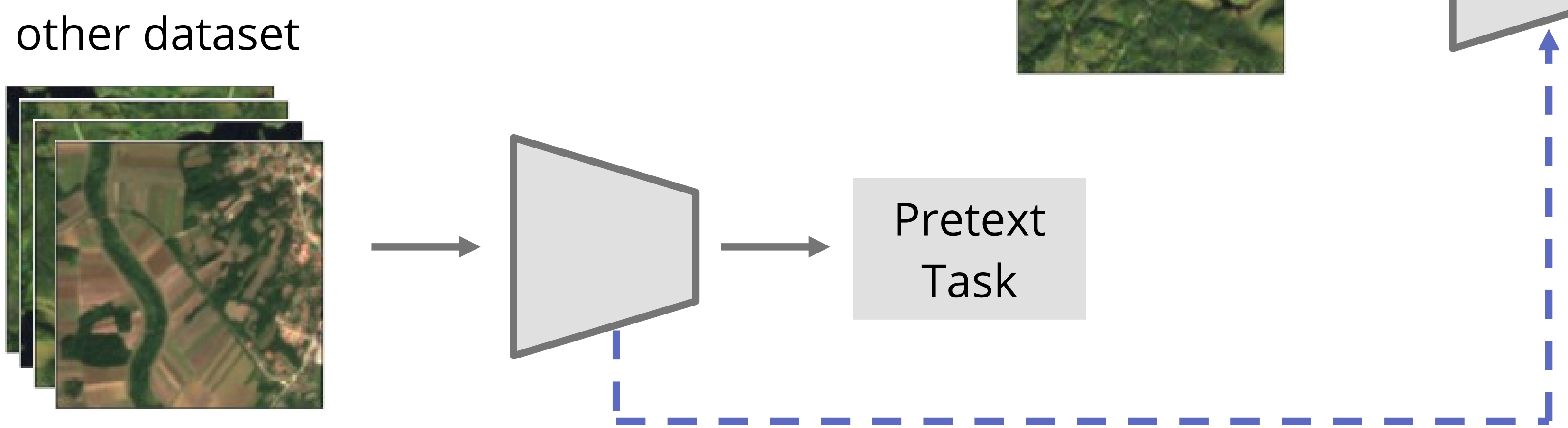


We initialize our model with the **pre-trained** model weights; training starts not from scratch!

How can we use annotated data more efficiently?

- Data augmentations
- Data Fusion
- Multi-task Learning
- Transfer Learning
- Self-supervised Learning





In SSL, we pretrain our model in a **self-supervised way** (no labels required) and then apply transfer learning to learn our actual task more efficiently.

How can we use annotated data more efficiently?

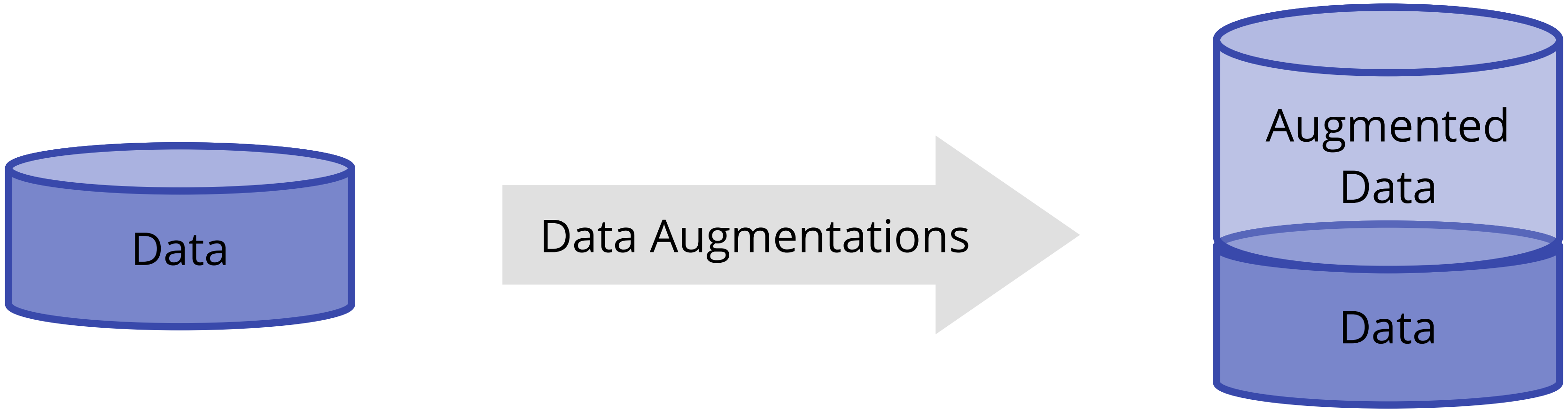
- Data augmentations
- Data Fusion
- Multi-task Learning
- Transfer Learning
- Self-supervised Learning

We will introduce these methods in the following and implement some them using PyTorch after the coffee break.

Data Augmentations

Data Augmentations

Data augmentations are a means to synthetically “increase the size” of your dataset. Augmentations are **transformations** that affect input data but not the corresponding labels; as a result, models trained with data augmentations tend to be **more robust** and **less prone to overfitting**.



Data Augmentations in Computer Vision

Original

Flip

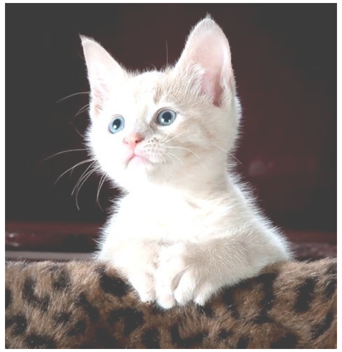
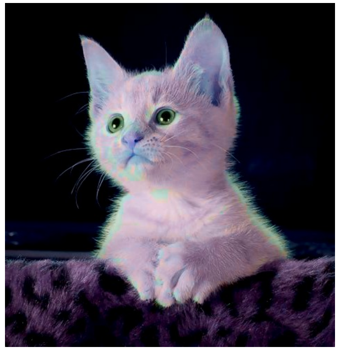


Image Enhancements



Color distortions

Crop

Data Augmentations for Remote Imaging Data

Original

Flip

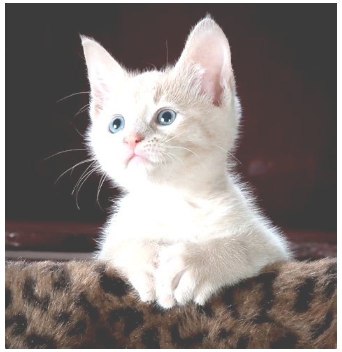
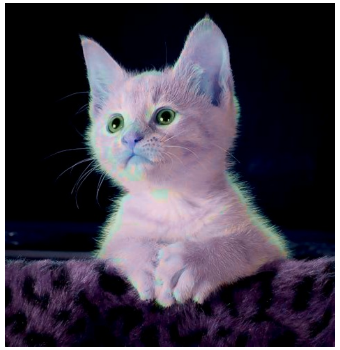


Image Enhancements



Color distortions

Crop

+ Rotations!

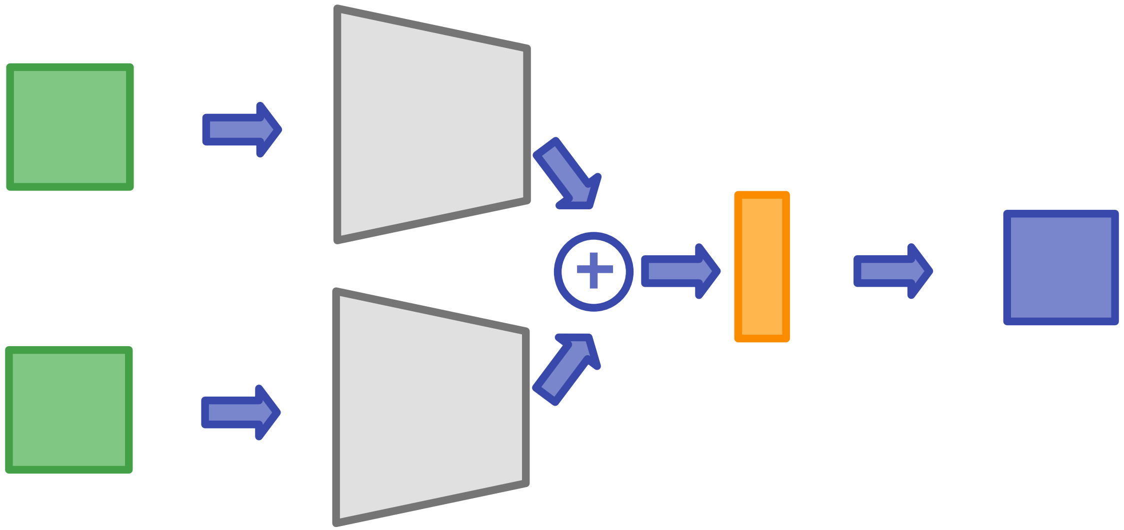
Data Augmentations for Remote Sensing

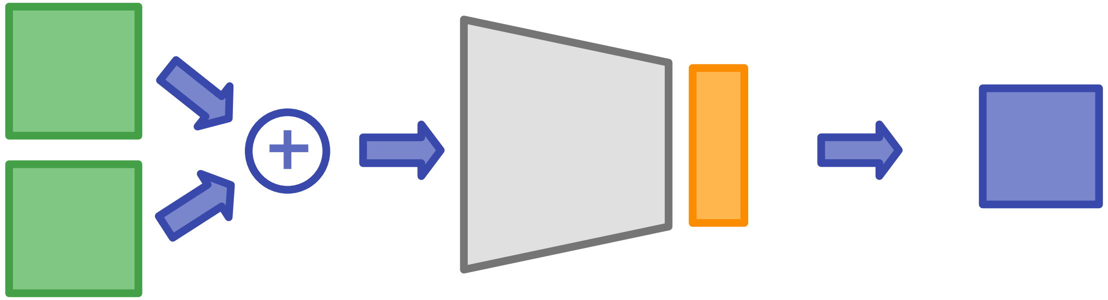
Data augmentations are a powerful method, but they have to be used with care: some transformations might be unphysical and harm/confuse the model.

If used properly, there is no disadvantage in using data augmentations.

Data augmentations are generally easy to implement, which is why we will not look at them in more detail...

Data Fusion



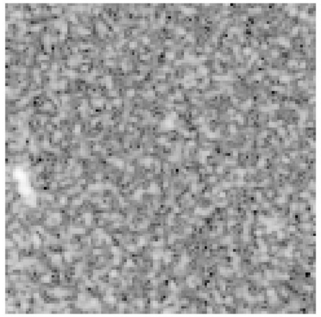


Data Fusion is a technique in which different data modalities are combined (“fused” fusion is to better perform a task by combining relevant data.

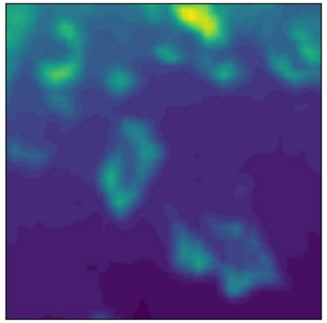
Earth observation is predestined for Data Fusion, as EO sensors colle erent data modalities:

Paolo Gamba will present Data Fusion for change detection in urban areas tomorrow!

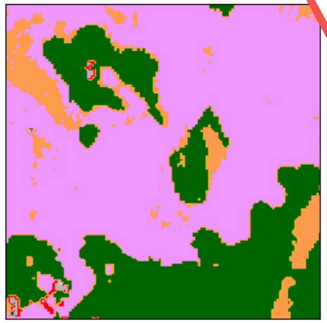
Multispectral
(e.g., Sentinel-2, Landsat)



SAR
(e.g., Sentinel-1, ICEye)



DEM
(e.g., Copernicus DEM)



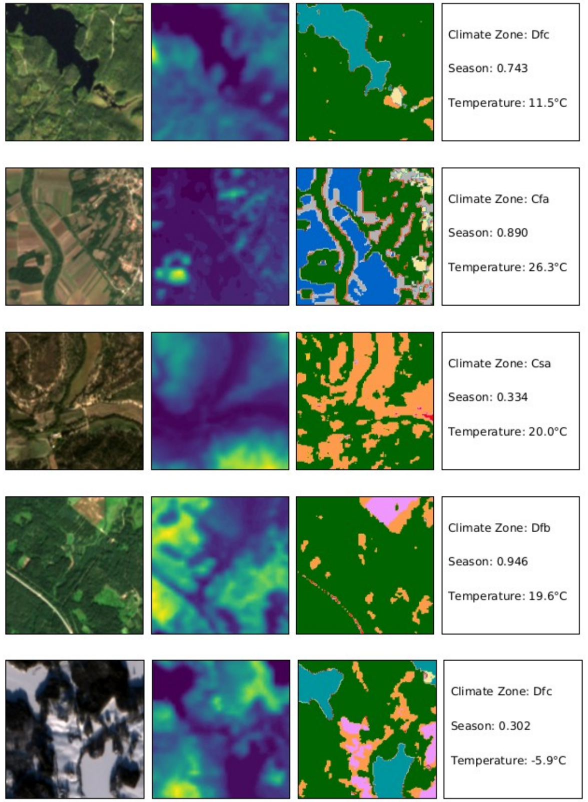
LU/LC
(e.g., Corine, Esa WorldCover)

?

Meta Data
(e.g., weather data, observation circumstances)

ben-ge: a truly multimodel dataset for EO

To explore the use of multimodal for Data Fusion (and other methods), we will use a specifically designed dataset:



BigEarthNet contains 590,326 patches of co-located Sentinel-1/2 data.

ben-ge extends BigEarthNet by the following data modalities:

- Elevation data (Copernicus DEM GLO-30)
- Land-use/land-cover maps (ESA Worldcover)
- Environmental data (ERA-5)
- Climate zone classification (Beck et al. 2018)
- Seasonal encoding

ben-ge serves as a testbed for combining different EO data modalities. For more details, check out https://github.com/HSG-AIML/ben-ge

We will use a subset of ben-ge, ben-ge-800, in this tutorial.

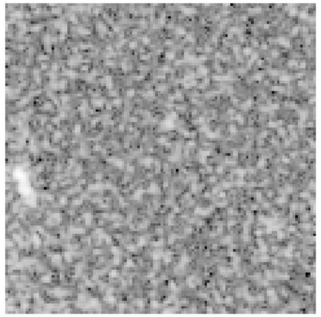
ben-ge: a truly multimodel dataset for EO

What data modalities are available in ben-ge?

BigEarthNet-MM

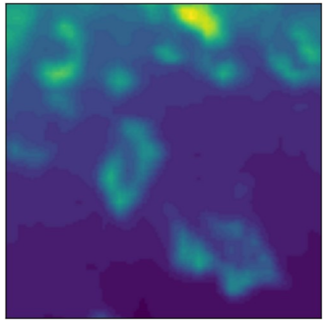
Sentinel-2
Multispectral

12 bands
Level-2A

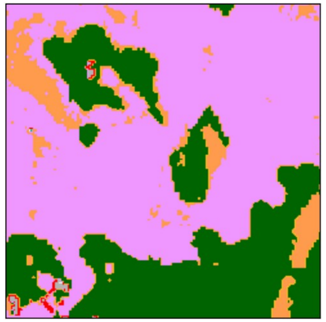


Sentinel-1
SAR

2 bands



Copernicus
DEM
(GLO-30,
resampled)



ESA WorldCover
LU/LC

8/11 classes

?

Meta Data

ERA-5 weather
Climate zones
Seasonality

10m resolution

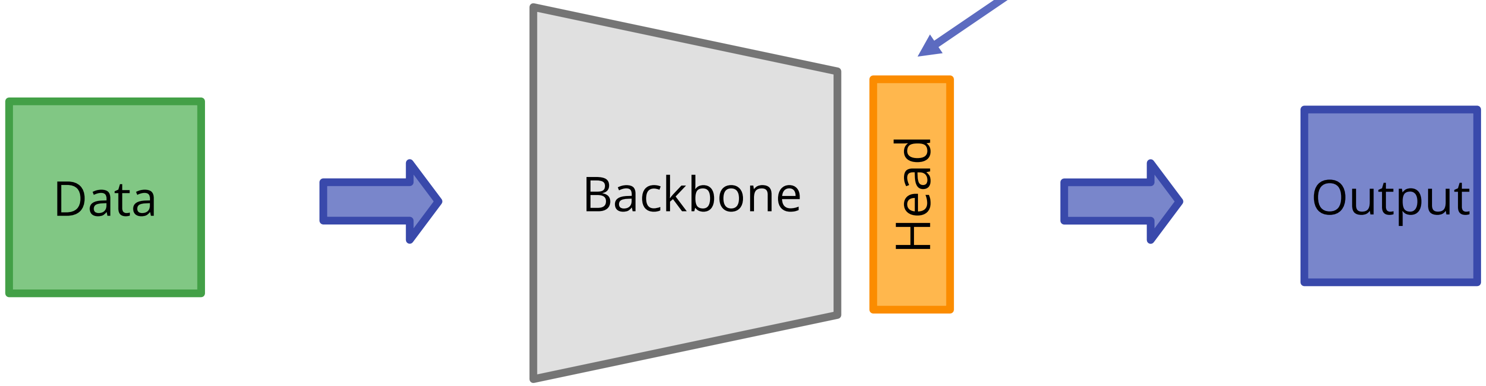
Data Fusion for Deep Learning

How can we leverage Data Fusion in Deep Learning?

Default supervised learning setup

Feature extractor
(in this tutorial: U-Net)

“Head” predicts the target based on the learned representations (features); e.g., classification head.

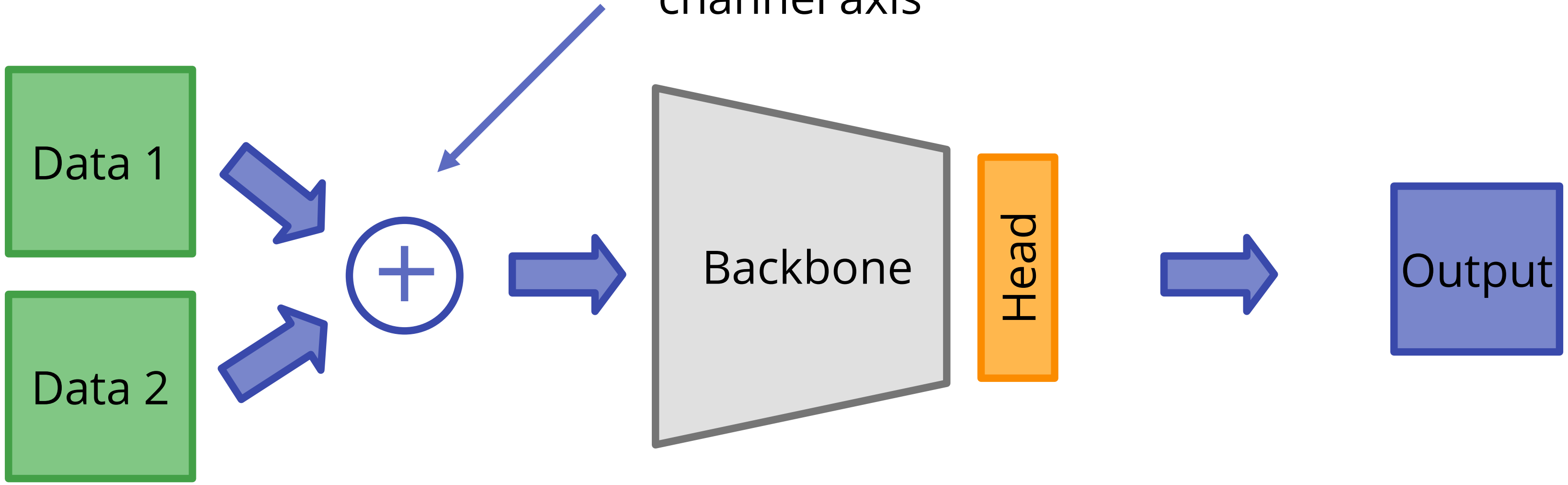


“Default supervised learning setup”

Early Fusion

In Early Fusion, two (or more) data modalities are combined before they enter the backbone:

Concatenation: stack data (all of the same shape) along the channel axis

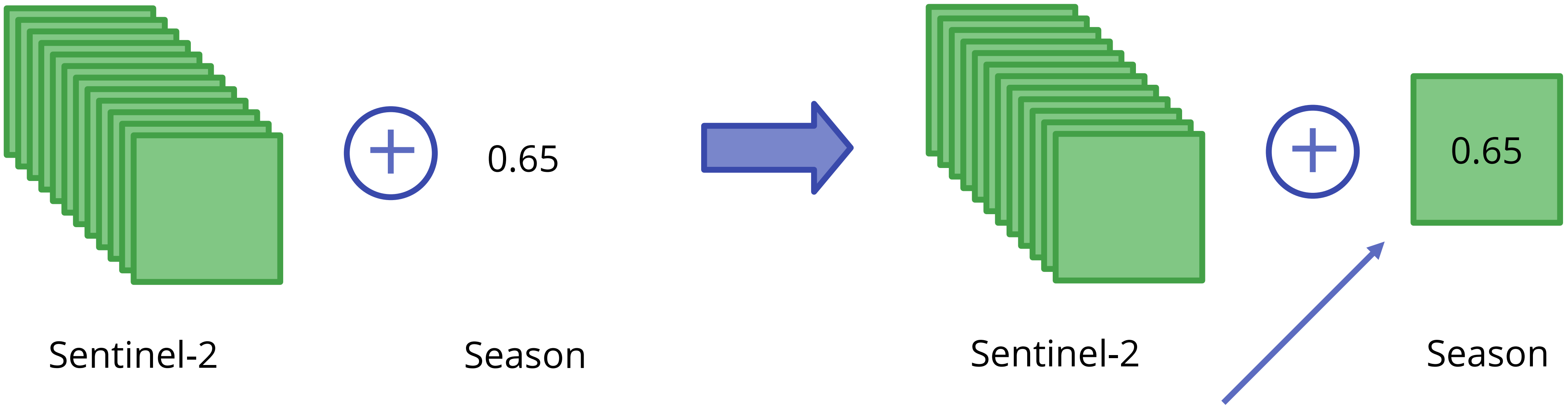


Different
data modalities

Early Fusion: Different Data Shapes

Early Fusion is simple if the data modalities to be combined have the same shape (e.g., map-like features with the same extent).

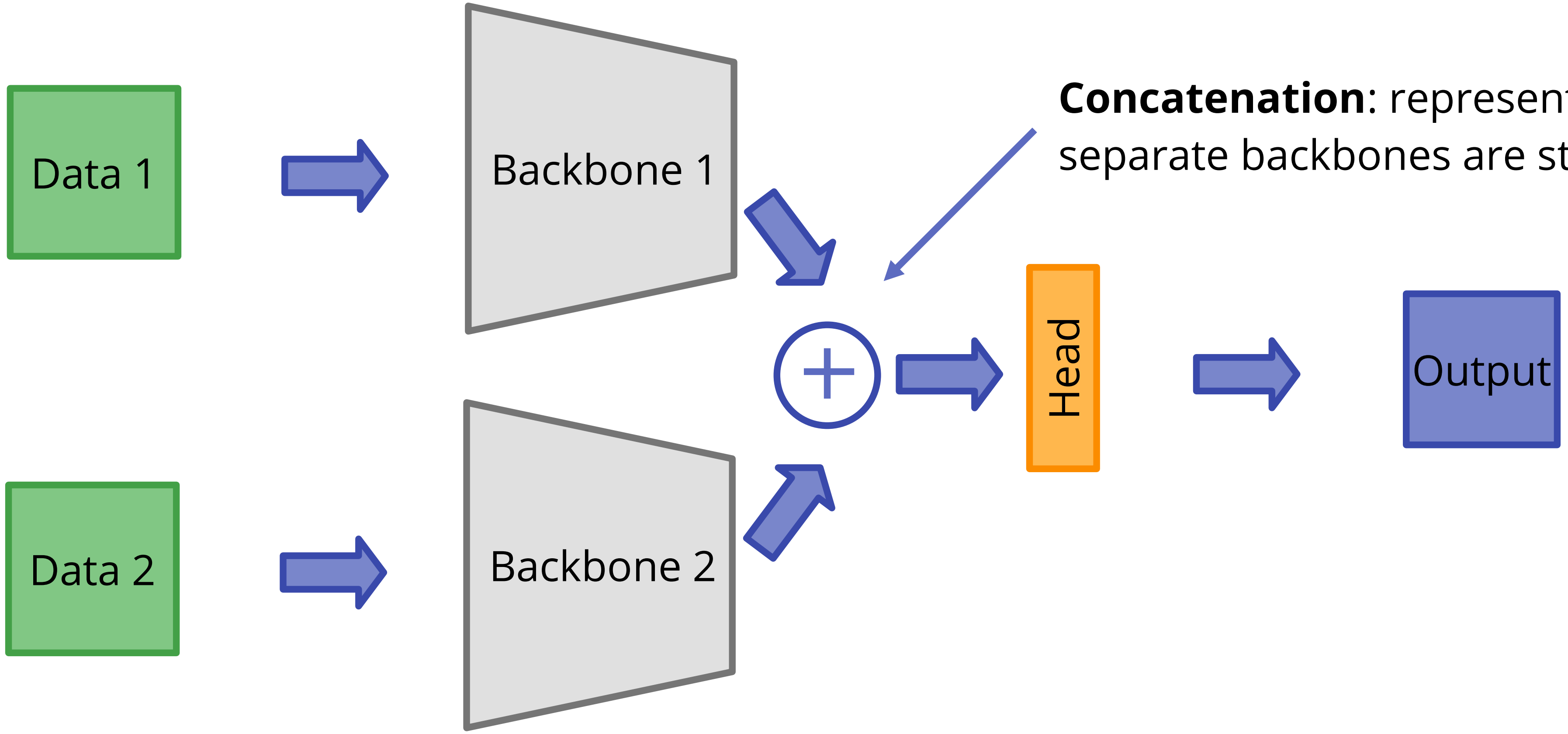
But: how to combine Sentinel-2 data (12 channels x 120 px x 120 px) with patch-global seasonality (scalar value in the range [0, 1]) data?



Blow-up patch: same height and width as Sentinel-2; each “pixel” equals the global value (0.65)

Late Fusion

In Late Fusion, two (or more) data modalities are combined after passing through separate backbones:



Concatenation: representations emerging from separate backbones are stacked along either axis

Backbones might be completely separate, or have shared weights.

Let’s implement some Data Fusion techniques into our model!

Data Fusion: An example

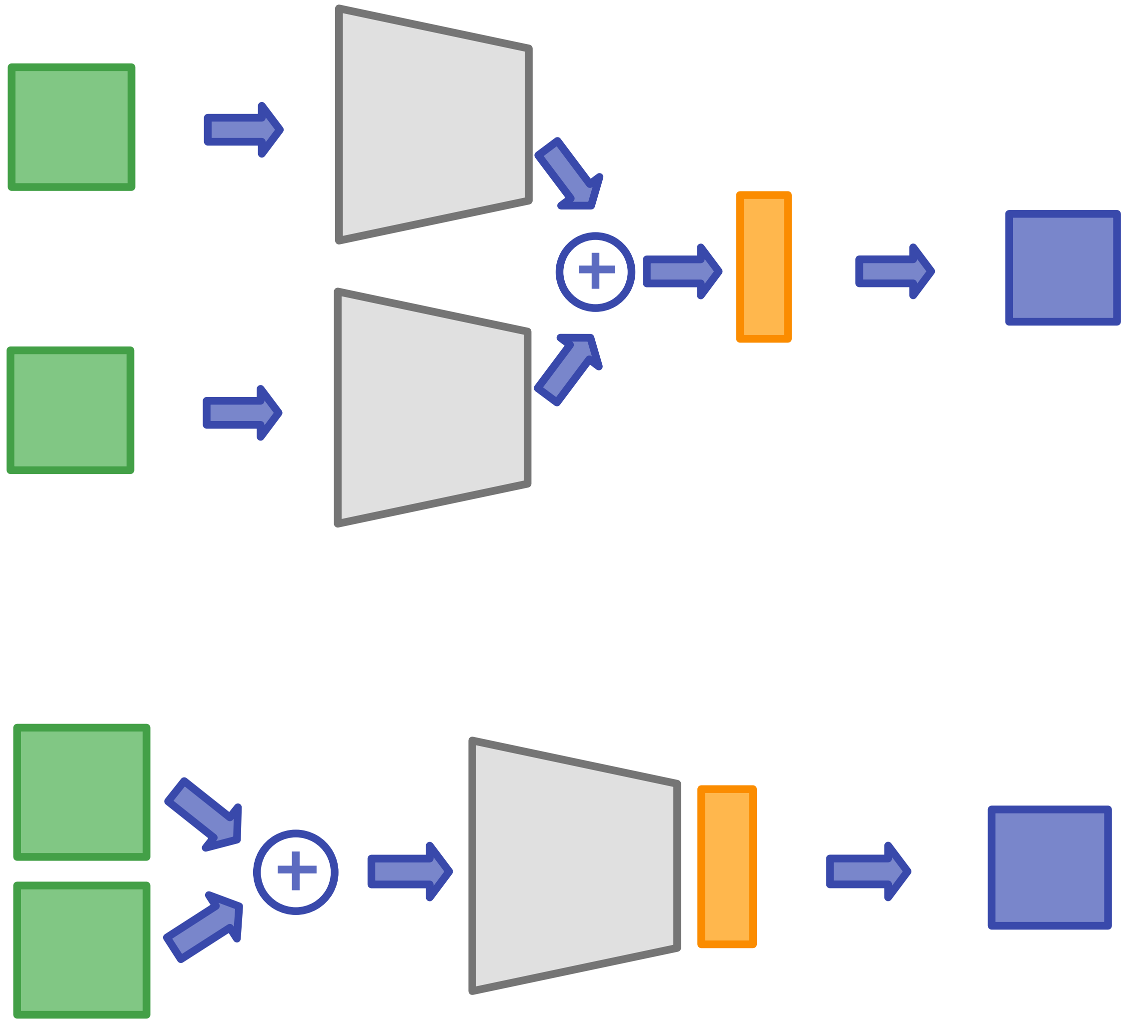
Which data modalities make sense to combine? (Mommert et al. 2023)

N	Sen-2	Sen-1	Climate	DEM	Weather	Season	Classification [%] F1-score	Classification [%] Accuracy	Segmentation [%] IoU	Segmentation [%] Accuracy
1	✓						**77.12**±0.64	96.21±0.08	**39.17**±0.09	87.57±0.05
		✓					73.09±0.24	95.60±0.05	31.70±0.17	82.65±0.05
			✓				70.50±0.34	94.69±0.03	14.70±0.32	60.65±1.35
				✓			55.96±1.00	93.53±0.15	26.25±0.48	76.92±0.63
					✓		46.15±0.68	91.60±0.02	6.30±0.05	45.20±0.08
						✓	39.15±0.74	91.75±0.05	6.01±0.34	43.89±0.51
2	✓	✓					**82.81**±0.29	97.03±0.04	**39.67**±0.16	87.98±0.07
	✓					✓	78.61±0.67	96.42±0.08	38.92±0.21	87.37±0.10
3	✓	✓	✓				**85.12**±0.34	97.39±0.05	39.63±0.23	87.94±0.12
	✓	✓		✓			83.30±0.43	97.10±0.08	**39.71**±0.21	88.05±0.11
	✓	✓				✓	—	—	39.61±0.19	87.93±0.12

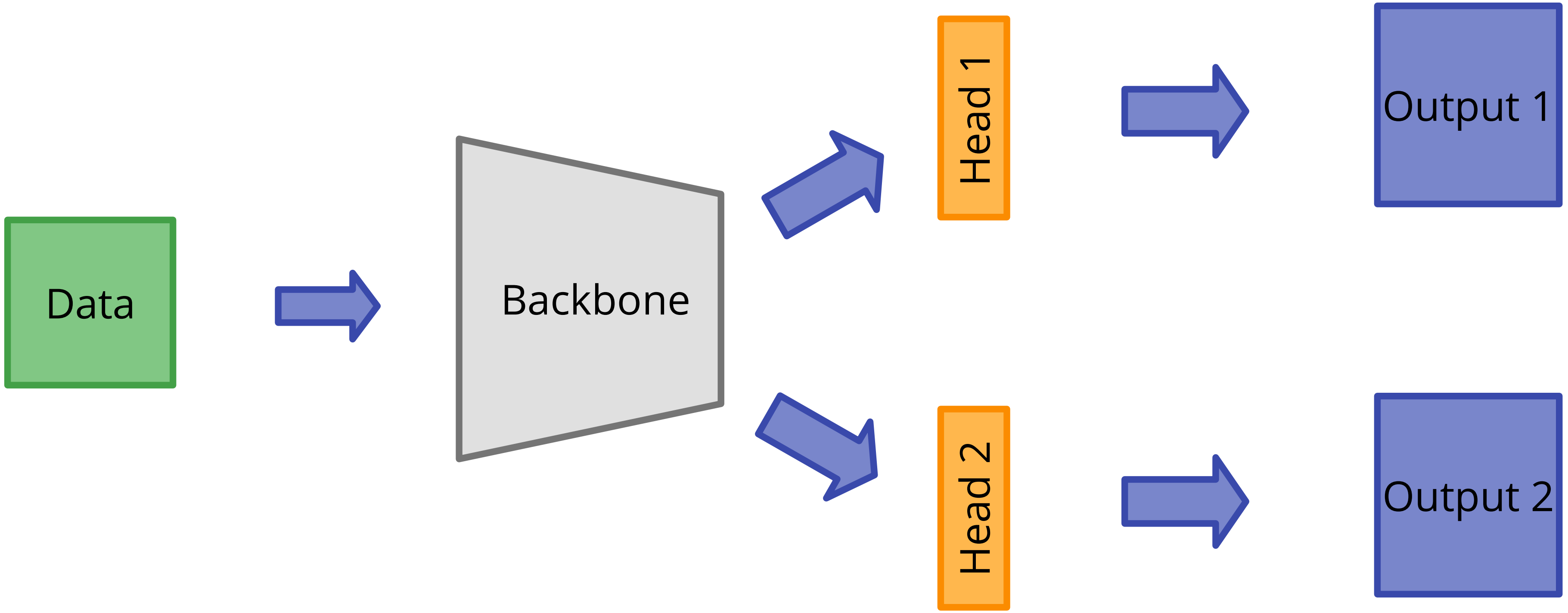
… it depends on the downstream task and the data…

Which method is better: in most cases, late fusion seems to be more beneficial (might be a fallacy).

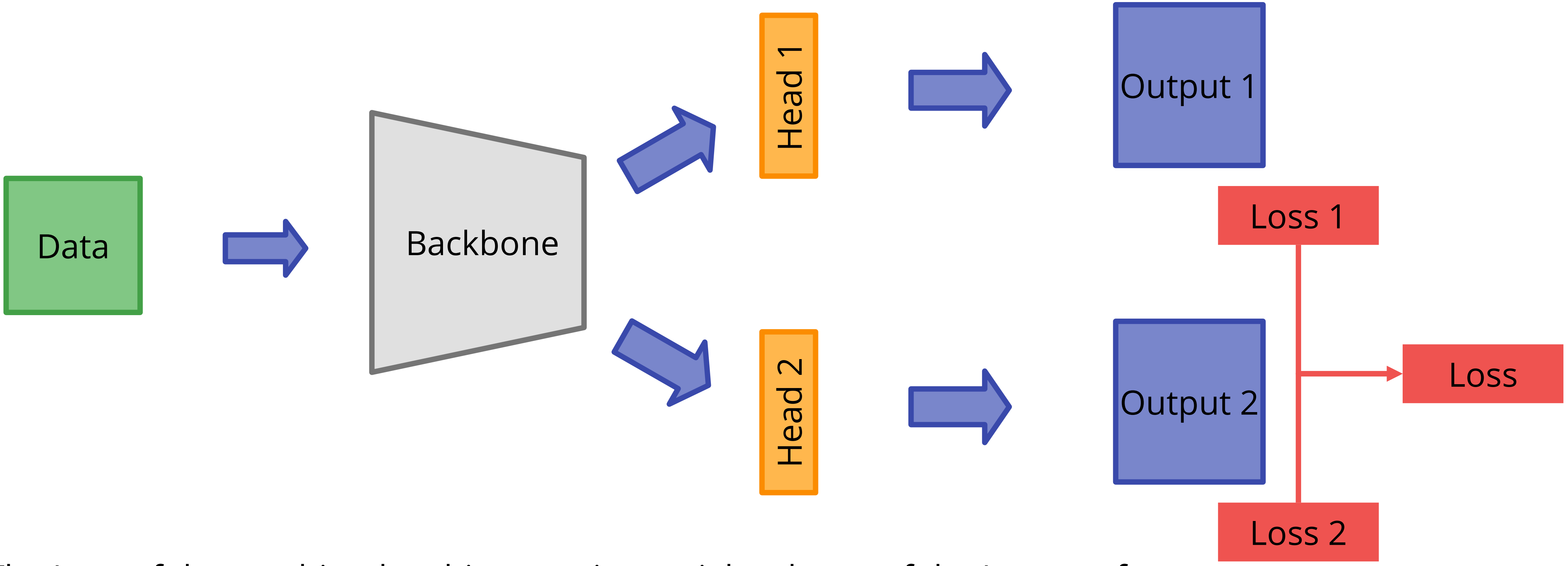
Multitask Learning



Multitask Learning



How are multitask architectures trained?



The Loss of the combined architecture is a weighted sum of the Losses of the individual downstream tasks.

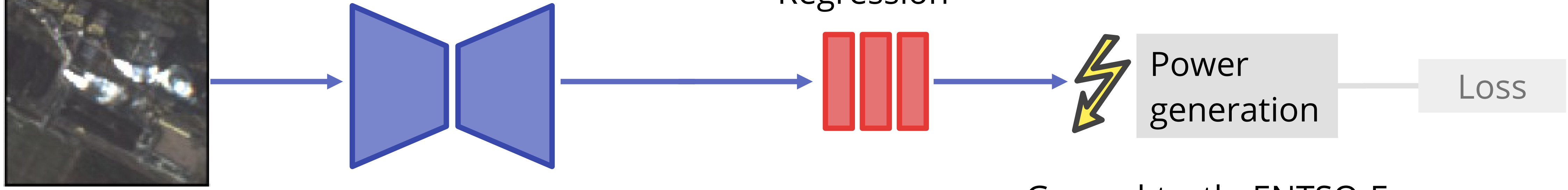
Let's implement some Multitask Learning techniques based on our model!

Multitask Learning: an Example

Idea: Can we train a neural network to estimate power and CO_2 output of power plants?

~3000 observations of
~150 power plants

Regression



Ground-truth: ENTSO-E

Sentinel-2
12 Bands

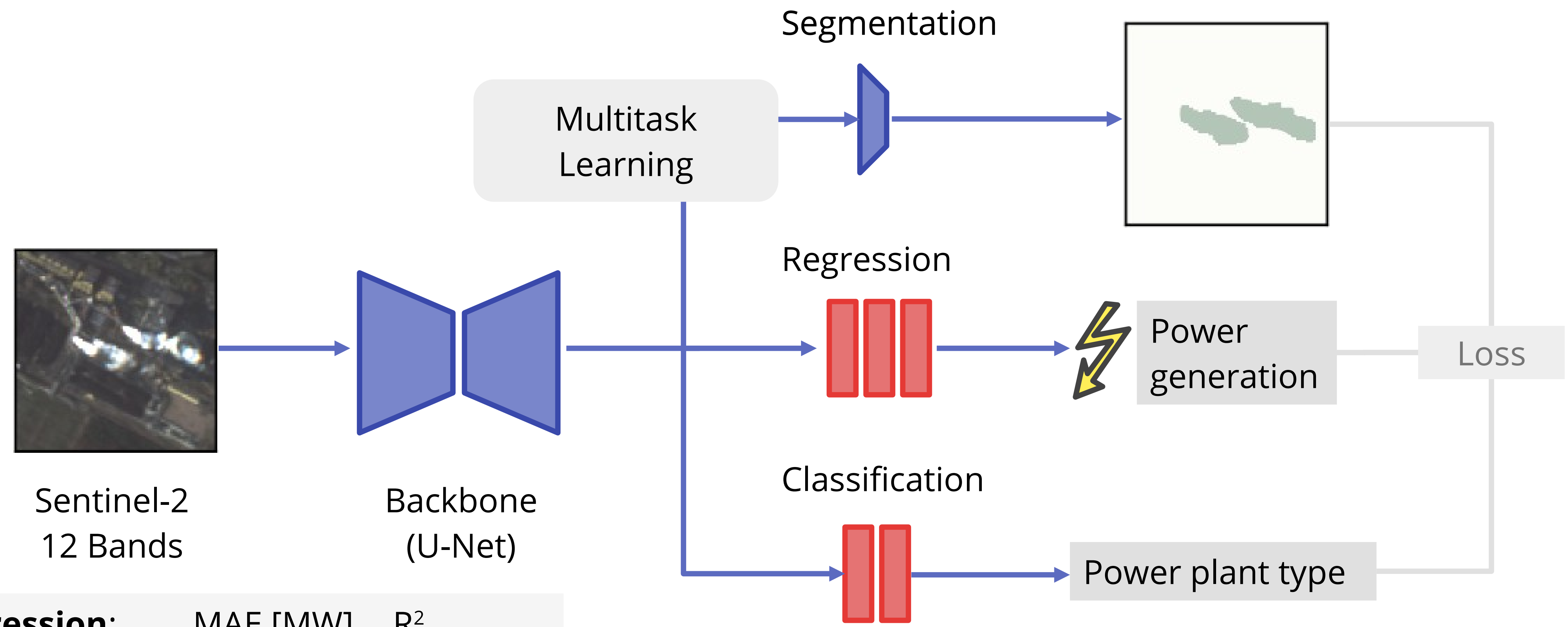
Backbone
(U-Net)

Regression:	MAE [MW]	R^2
Baseline:	202±20	65±5

Estimating power generation is possible. But can we improve it with Multitask learning?

Hanna et al. (2023)

Multitask Learning: an Example

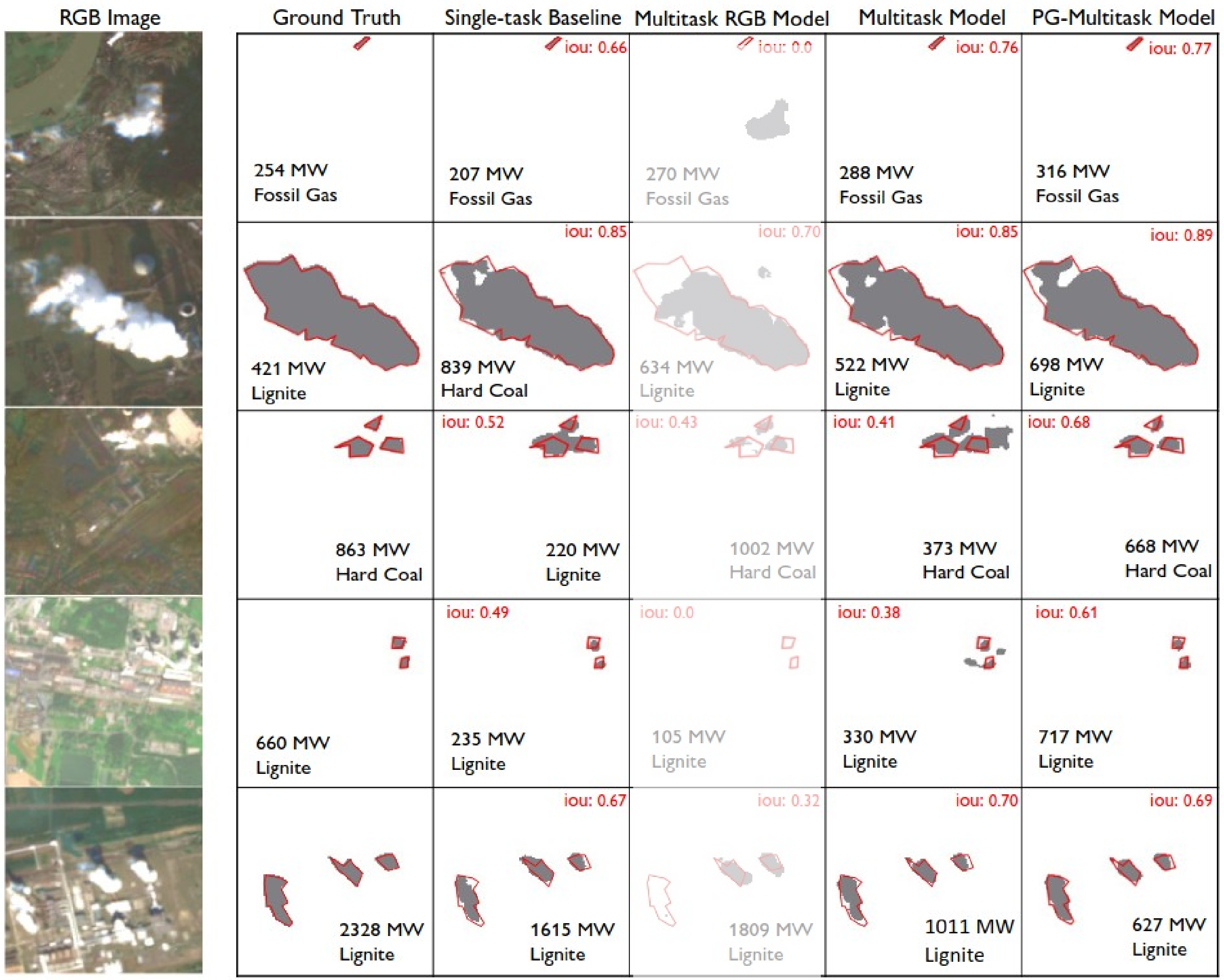


Regression:	MAE [MW]	R^2
Baseline:	202±20	65±5
Multitask:	187±4	66±6

Learning several tasks at the same time improves the performance significantly.

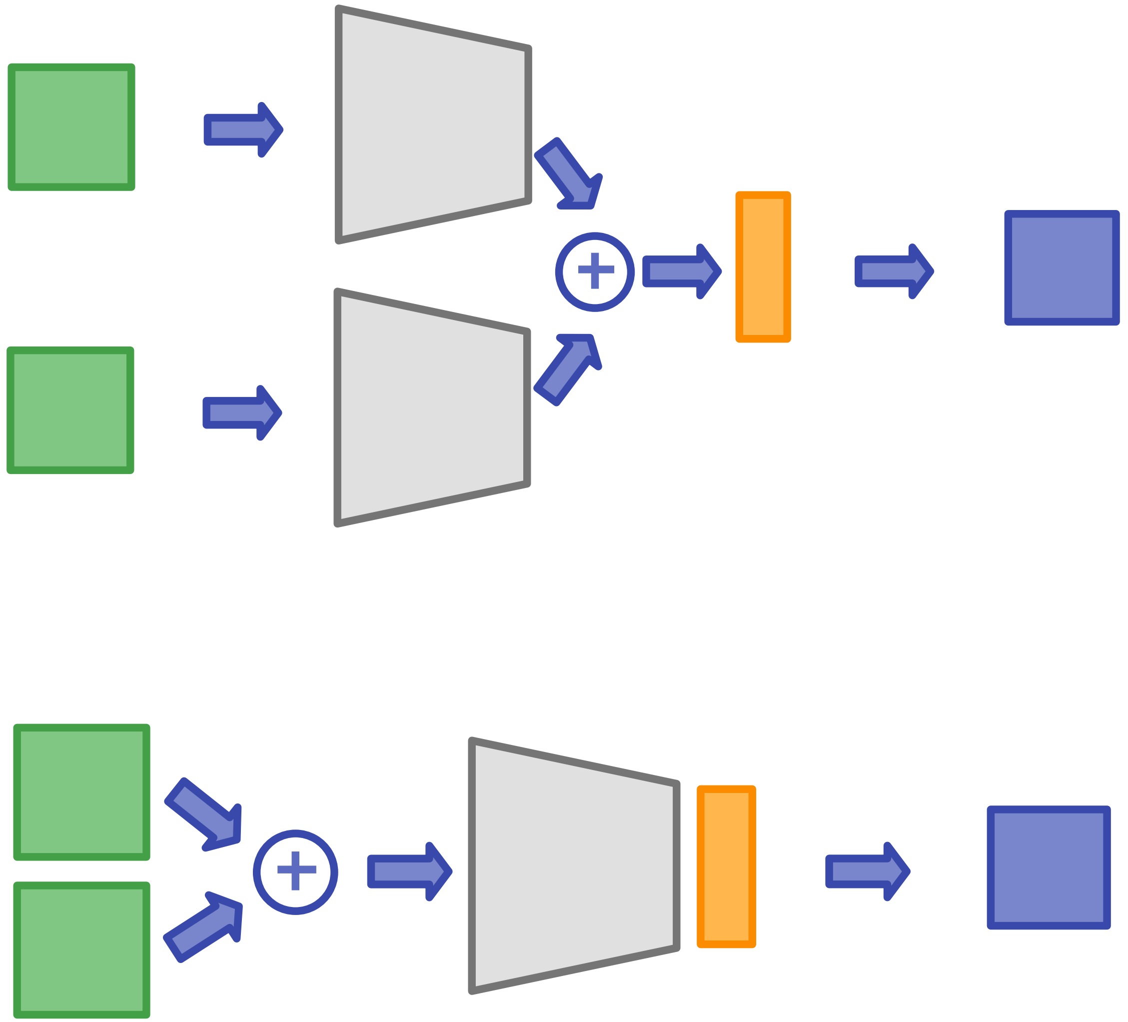
Hanna et al. (2023)

Multitask Learning: an Example

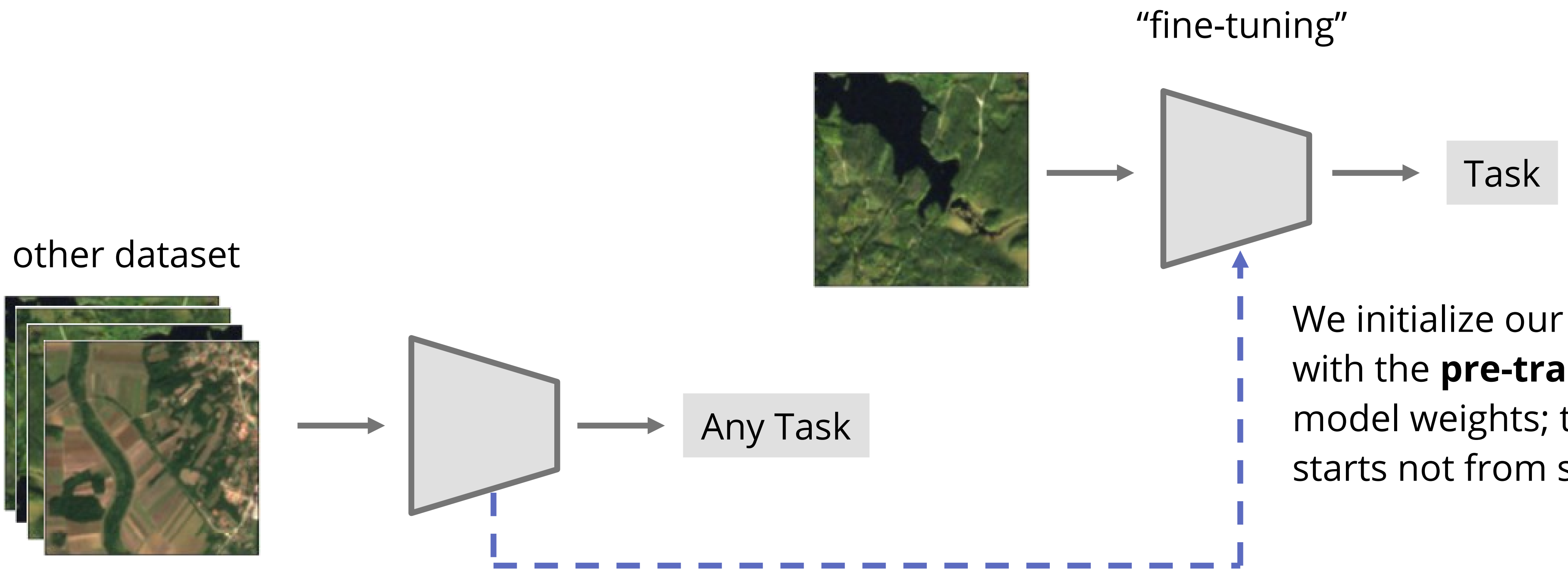


Hanna et al. (2023)

Transfer Learning and Self-supervised Learning



Transfer Learning

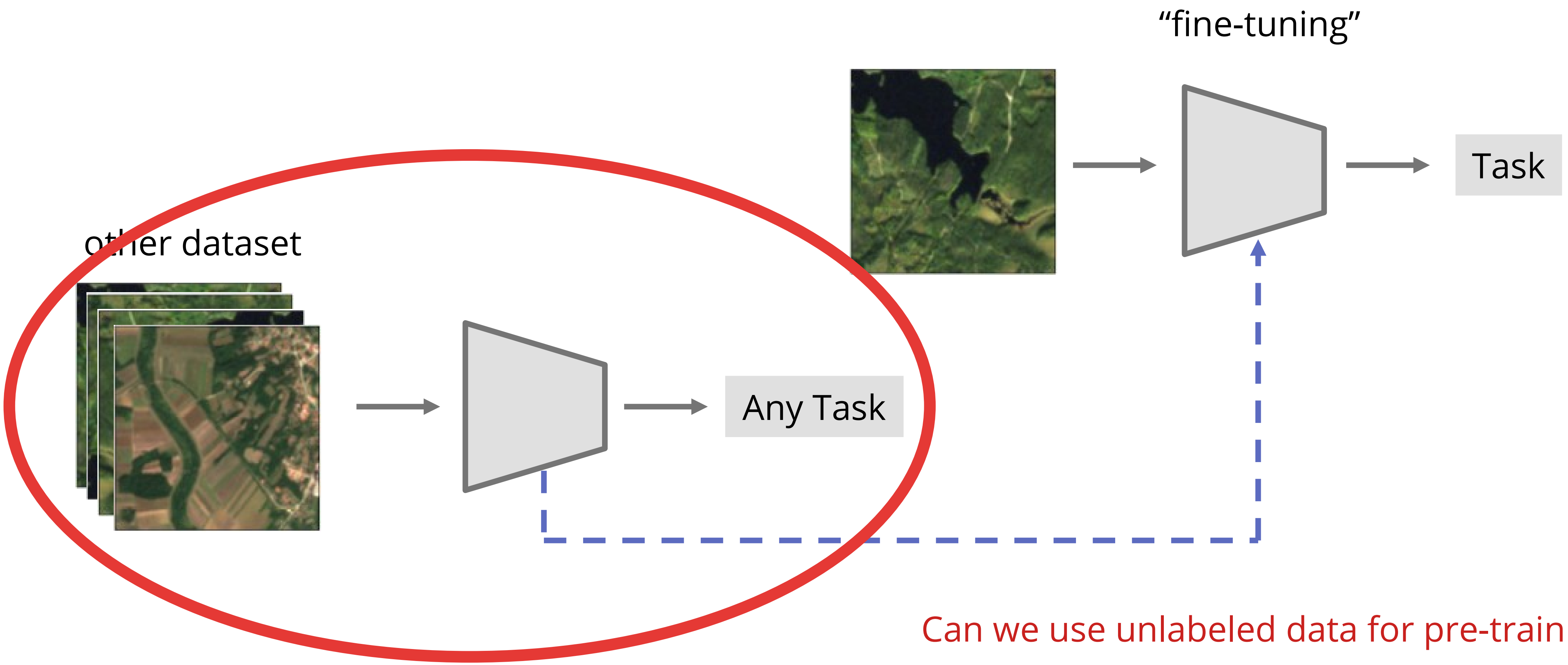


We initialize our model with the **pre-trained** model weights; training starts not from scratch!

In the end, transfer learning simply means that your model has previously been trained: you load a model checkpoint and resume training on your data and for your downstream task.

Implementing Transfer Learning is simple. Let’s do it!

Transfer Learning still needs labels



Can we use unlabeled data for pre-training?

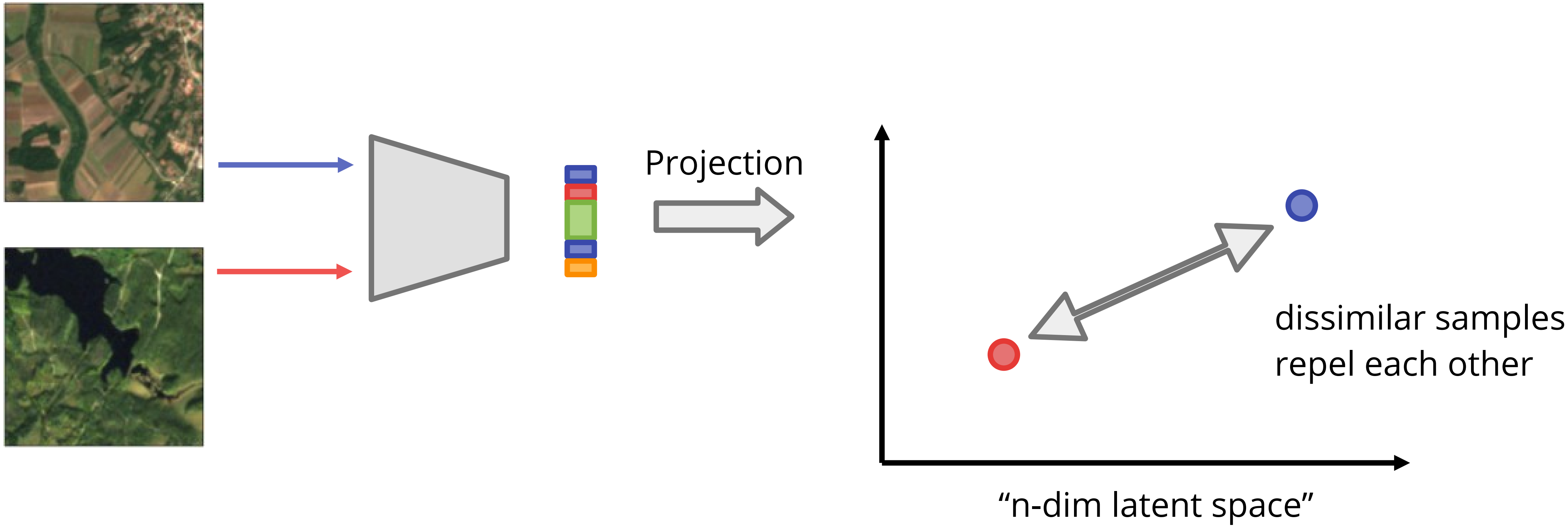
Self-Supervised Learning (SSL) and Transfer Learning

Self-supervised learning: learn “to see”, differentiate between image features (edges, colors) without supervision

Transfer learning: use the learned features to solve a task by providing “few labels”

Contrastive self-supervised learning

Contrastive learning setup (following SimCLR):

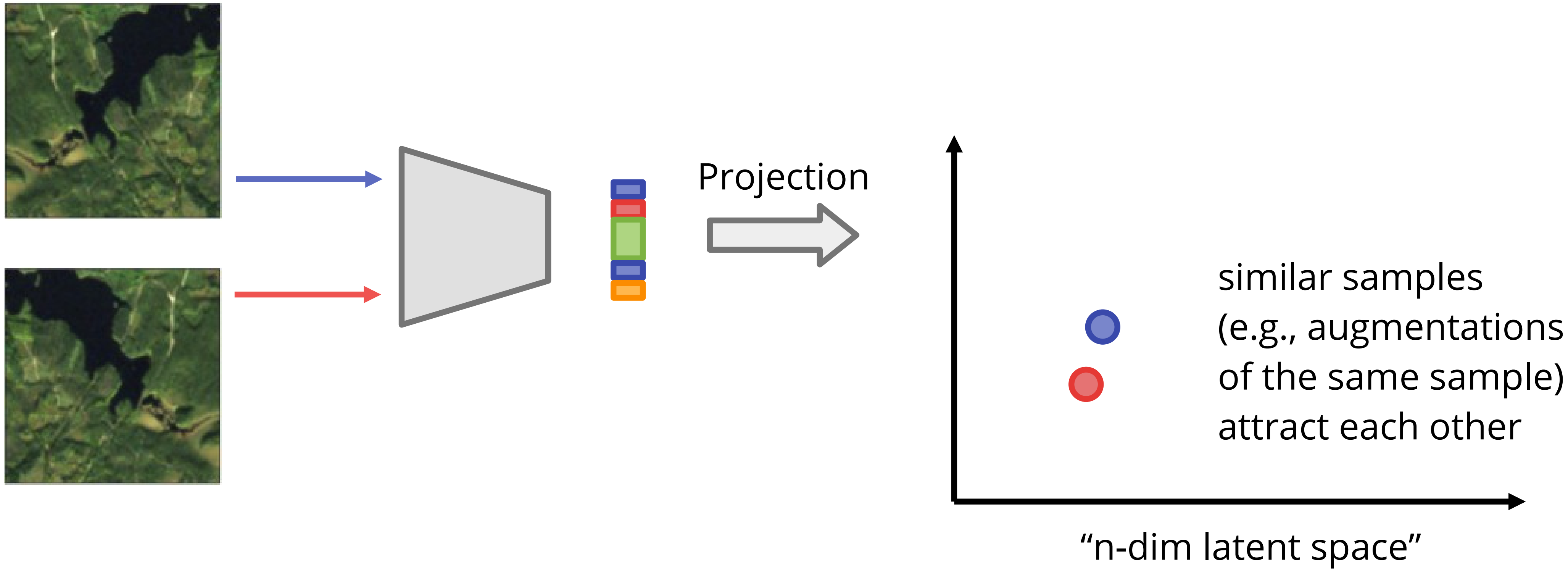


Chen, Ting, Simon Kornblith, Mohammad Norouzi and Geoffrey E. Hinton. "A Simple Framework for Contrastive Learning of Visual Representations." ArXiv abs/2002.05709 (2020)

Contrastive self-supervised learning

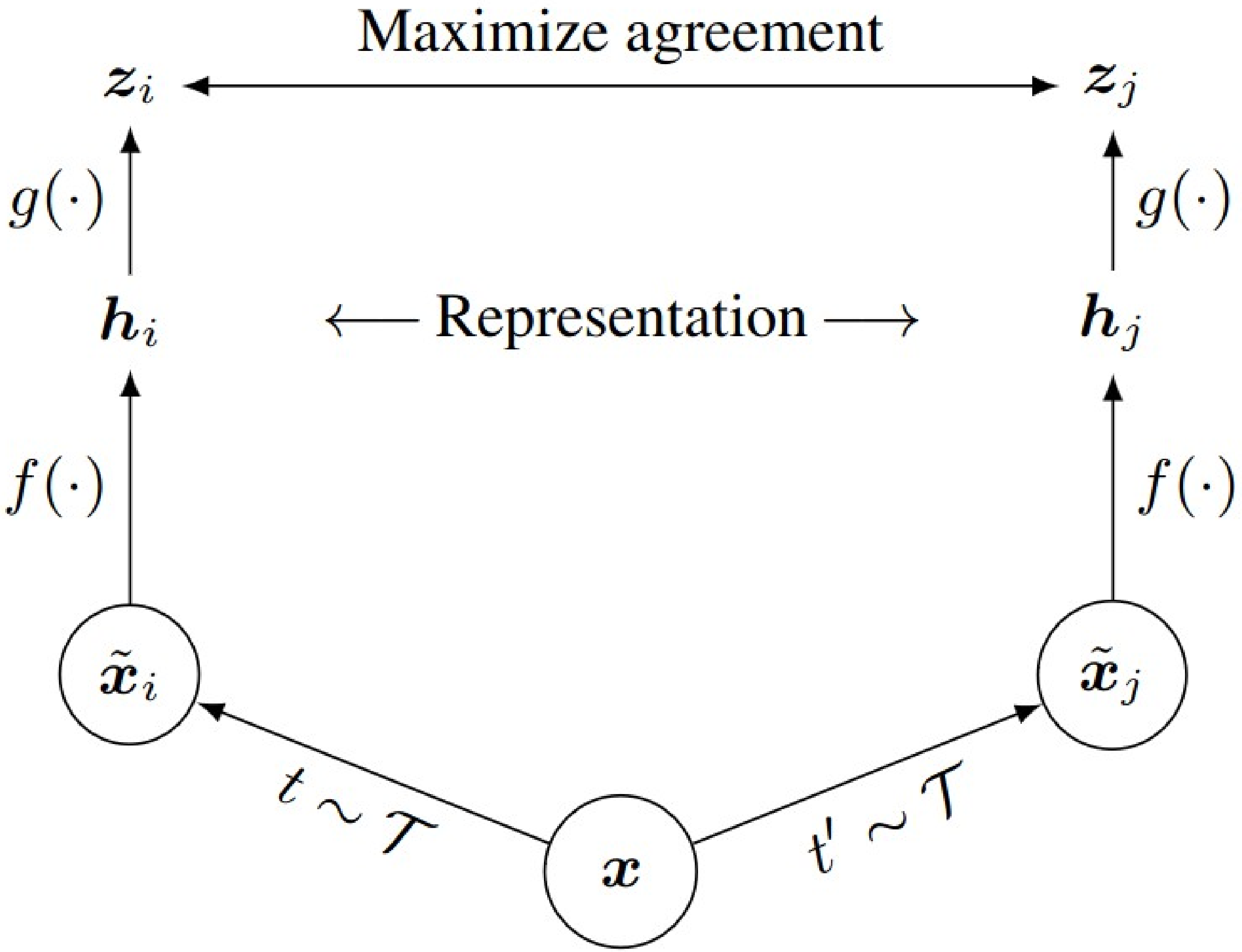
Yes we can! SSL is able to learn **rich representations** from large amounts of unannotated data.

Contrastive learning setup (following SimCLR):



Chen, Ting, Simon Kornblith, Mohammad Norouzi and Geoffrey E. Hinton. "A Simple Framework for Contrastive Learning of Visual Representations."
ArXiv abs/2002.05709 (2020)

SimCLR: A Simple Framework for Contrastive Learning of Visual Representations



Data augmentations (Transformations) are key for SimCLR to work.

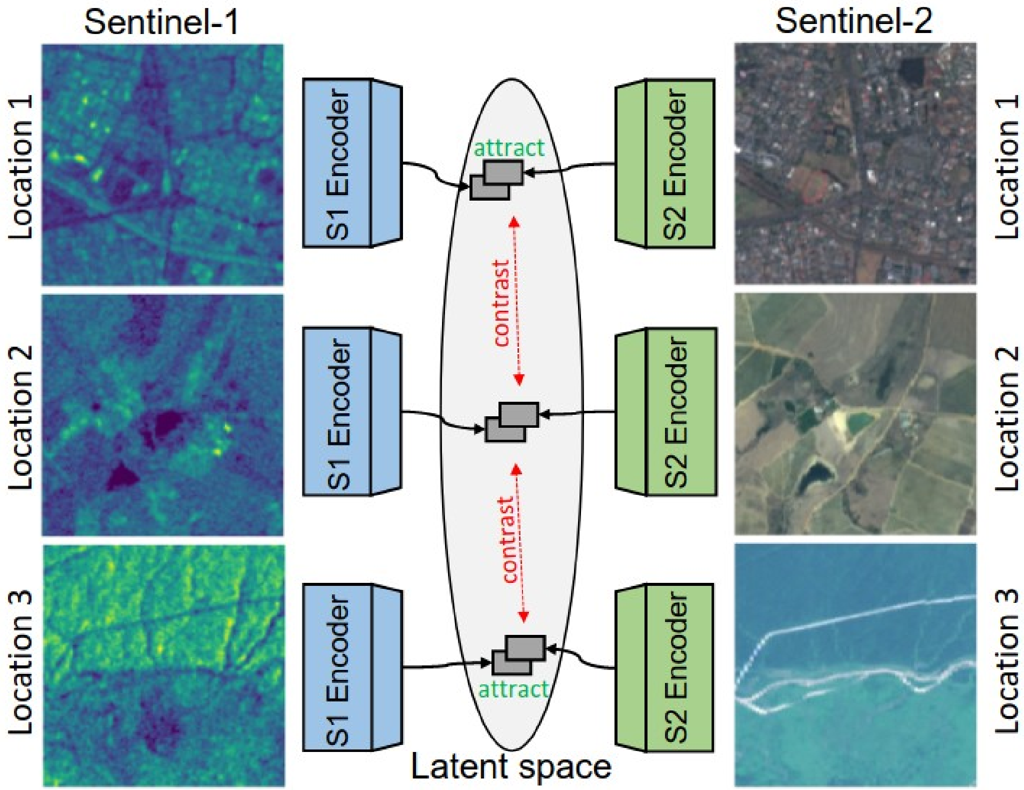
In remote sensing, we naturally have different views of the same scence (different times, different data modalities, etc.) We can leverage these views...

Chen et al. (2020)

Contrastive SSL for Earth observation: an Example

Pre-training

- Multimodal dataset (SEN12MS, ~181k Sentinel-1/2 patch pairs)
- Separate backbones for each modality
- Augmentation-free, contrastive setup



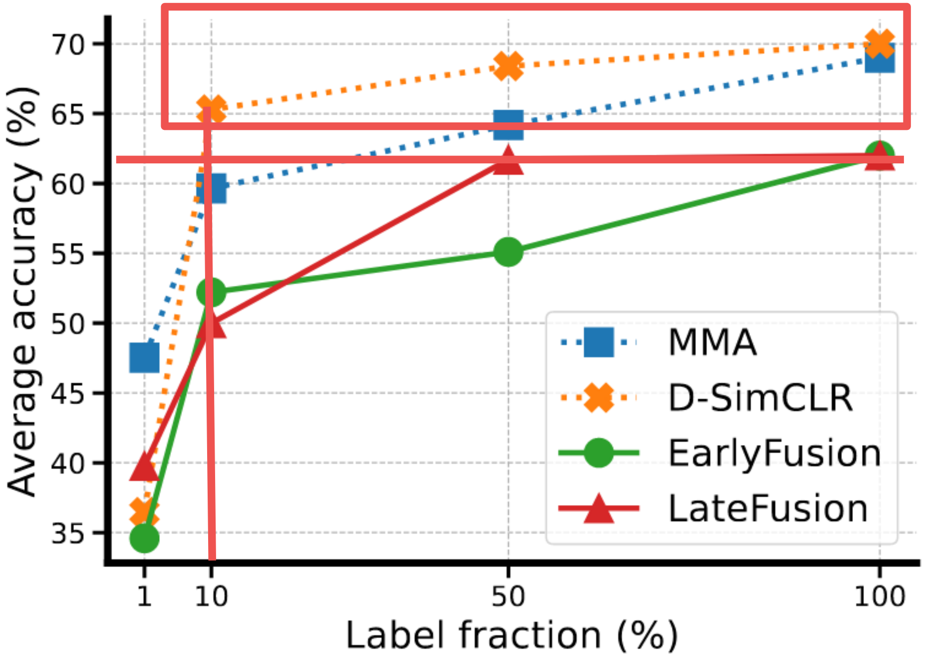
Scheibenreif, L., Mommert, M., Borth, D., “Contrastive Self-Supervised Data Fusion for Satellite Imagery”, ISPRS 2022.

Let’s see how we can build this architecture…

Contrastive SSL for Earth observation: an Example

Fine-tuning on classification task

- Annotations from DFC2020 high-res (10m) land use/land cover maps, ~5k patches, 8 classes
- Main result: pretrained models outperform supervised baselines with only **10% of training data**



Scheibenreif, L., Mommert, M., Borth, D., "Contrastive Self-Supervised Data Fusion for Satellite Imagery", ISPRS 2022.

This is it!

Summary

We introduced a number of methods to make more efficient use of labels (or use no labels at all):

- Data augmentations
- Data Fusion
- Multi-task Learning
- Transfer Learning
- Self-supervised Learning

Now go out into the world and use the code that we discussed for your own research!