LABORATOIRE
D'INFORMATIQUE
& SYSTÈMES
UMR 7020

Towards an unbiased detection of Galactic filaments using innovative Deep Learning methods

BERTHELOT Loris

LABORATOIRE D'ASTROPHYSIQUE DE MARSEILLE

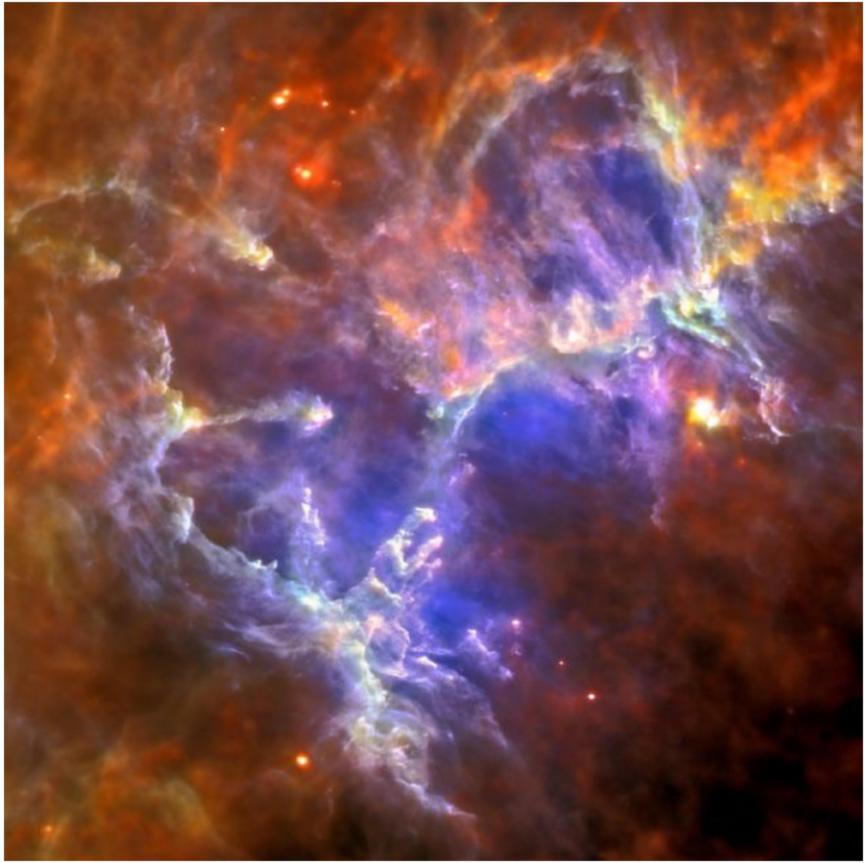
Outline

1. Introduction
2. Machine learning concepts
3. Dataset: Hi-GAL
4. Inference and learning on one single image
5. Normalization
6. Semi-supervised learning
7. PE-UNets: Adding physics information to improve performance
8. Improving PE-UNets with data augmentation
9. Cross-validation and model comparison
10. Applying our models to COHRS data ^{12}CO (3-2) survey

Introduction

Context

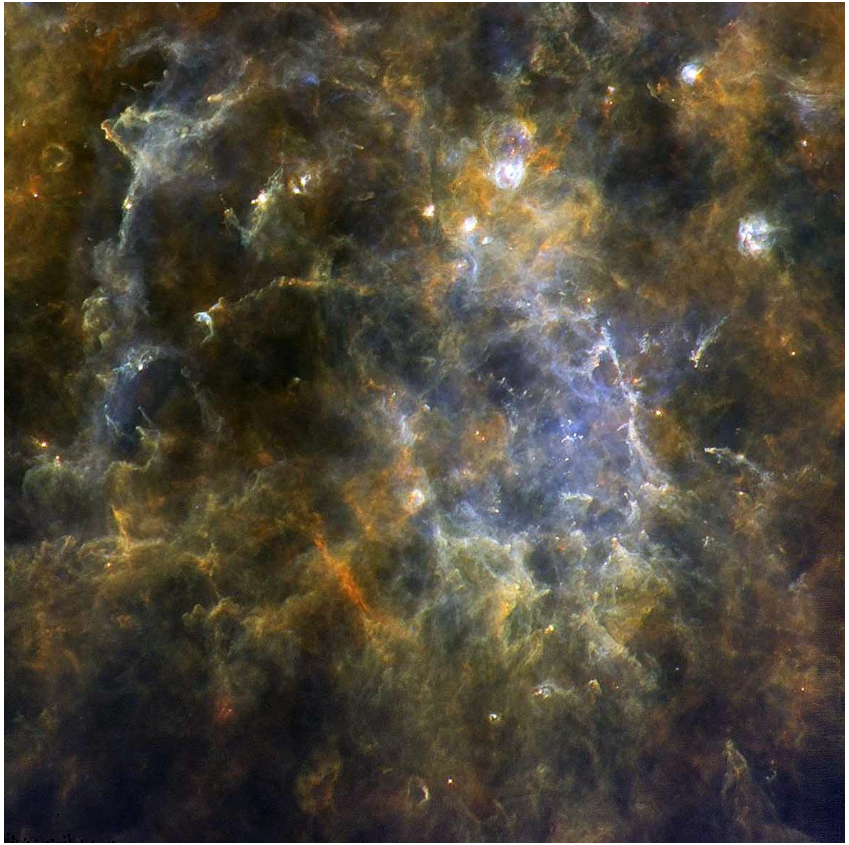
- Filaments are made of gas and dust
- Filaments host stars formation
- Need to detect filaments in an *unbiased* way to understand star formation



Eagle nebula (M16) with Herschel (70 160 250 mm), Xu+2019

What are filaments?

- Structuration of the interstellar matter
- Over-density compared to the surrounding medium
- Mechanisms affecting filaments:
  - Gravity
  - Magnetic field
  - Turbulence
  - Supernova feedback

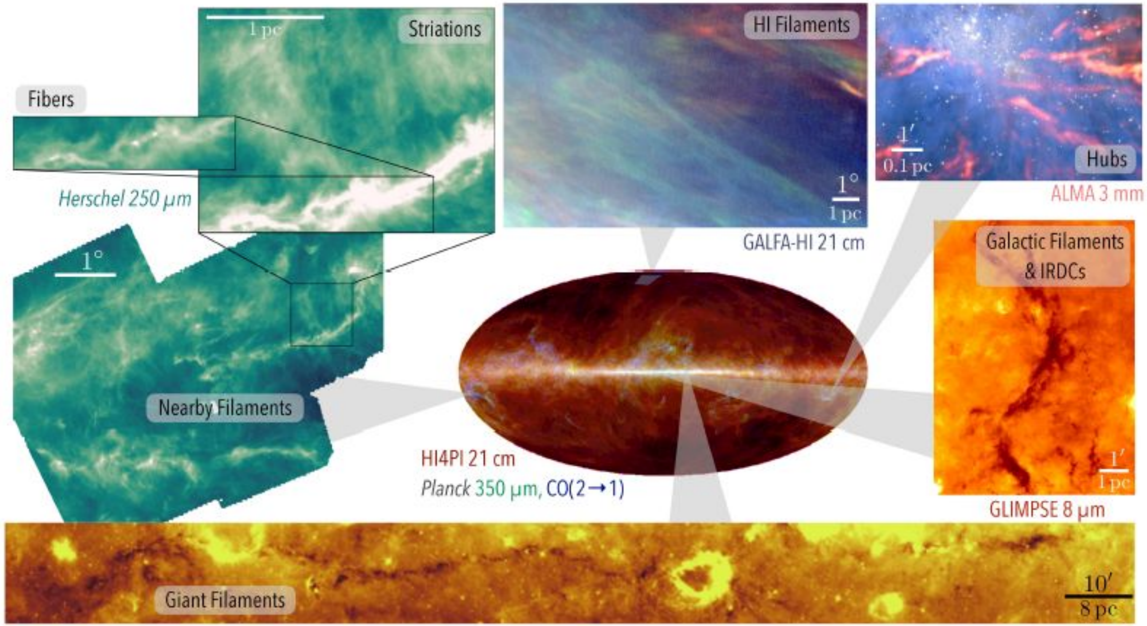


Herschel-Hi-GAL image of the Chamaeleon Galactic Star forming region
Credit: ESA/Herschel/PACS, SPIRE/Hi-GAL Project.

Filaments: A large diversity and a complex life cycle

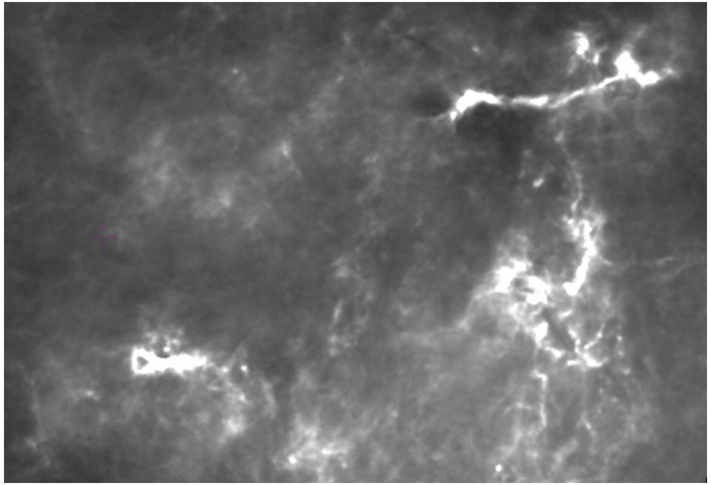
Filament properties:

- Shape
- Over-density level
- Orientation
- Length
- Width



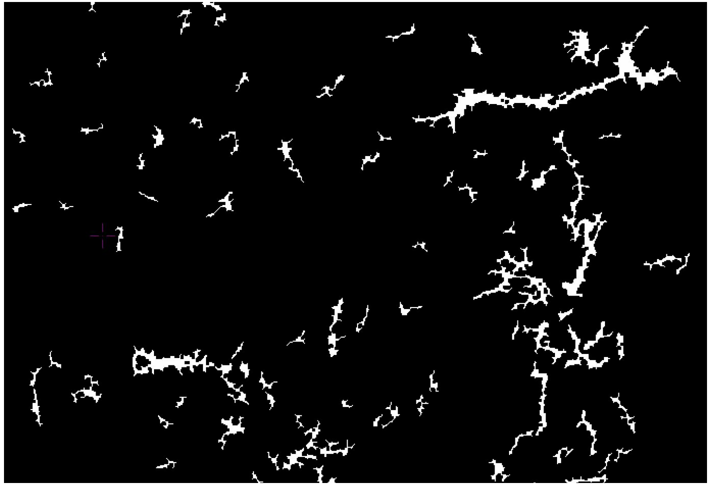
Hacar+2023, Protostars and Planets VII, 153

Objective: Detect filaments in an unbiased way



H_2 column density: Hi-GAL dataset, Molinari+ 2010

Filament detection



Hessian-based method, Schisano+2020

Existing methods

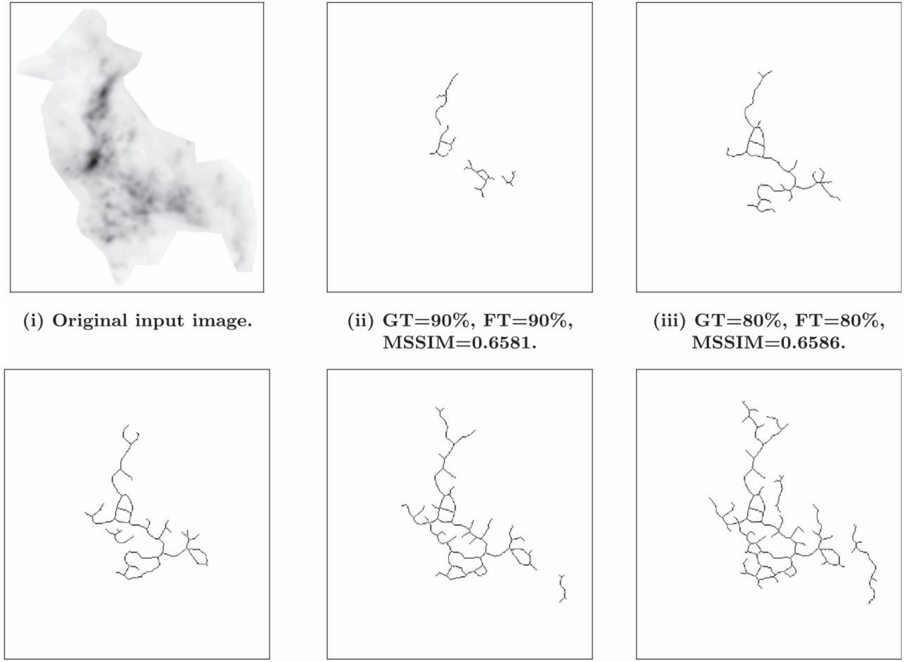
- Derivative-based approaches (e.g. DisPerSe, CRISPy)
- Pattern matching approaches (e.g. RHT, Filfinder, FilDReaMS)
- Multi-scale approaches (e.g. getfilaments, getsf, wavelet-based methods)

⚠ User hyper-parameters dependency

Hyperparameters issue

- Subjective results
- Incomplete extractions
- Over prediction algorithms
- Time consuming

Machine learning



(i) Original input image.

(ii) GT=90%, FT=90%, MSSIM=0.6581.

(iii) GT=80%, FT=80%, MSSIM=0.6586.

(iv) GT=70%, FT=70%, MSSIM=0.6589.

(v) GT=60%, FT=60%, MSSIM=0.6587.

(vi) GT=50%, FT=50%, MSSIM=0.6579.

Filfinder, Green+ 2017

Machine learning concepts

Train/Validation/Test setting

- Train our neural network

- Training control
- Choose best hyper-parameters

- Compute performances

Hyper-parameter selection

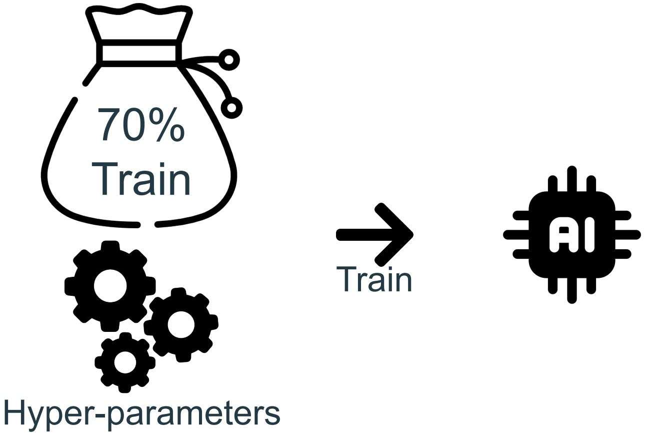
Hyper-parameter selection

Retrain and performance

Hyper-parameter selection

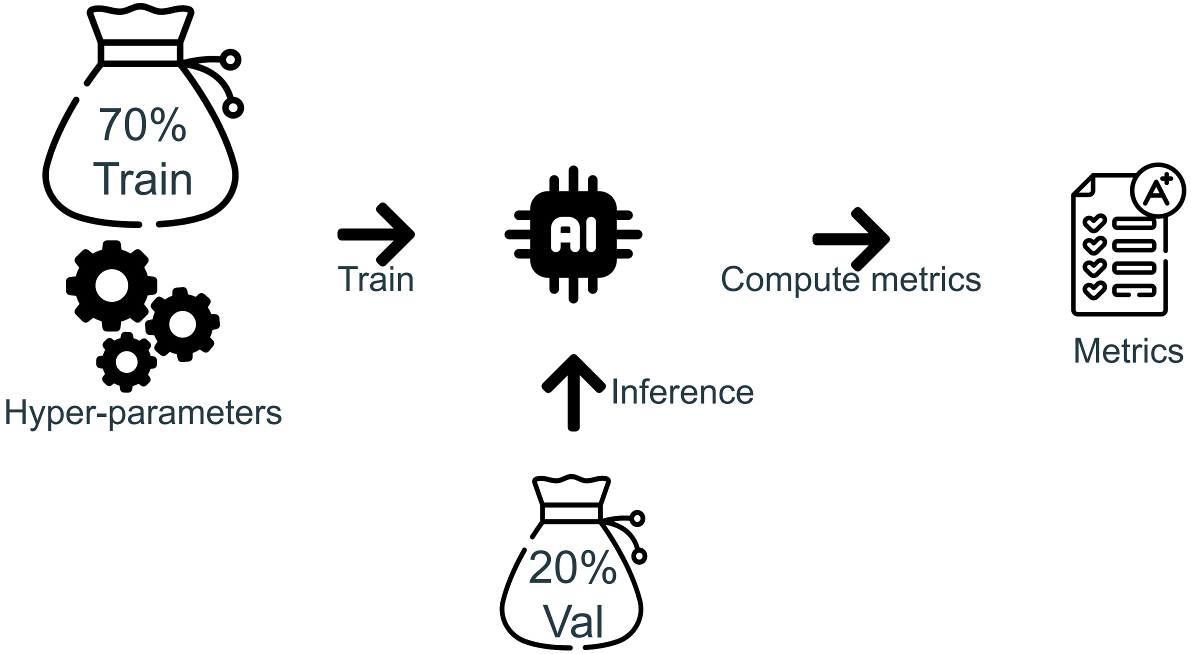
Hyper-parameter selection

Retrain and performance



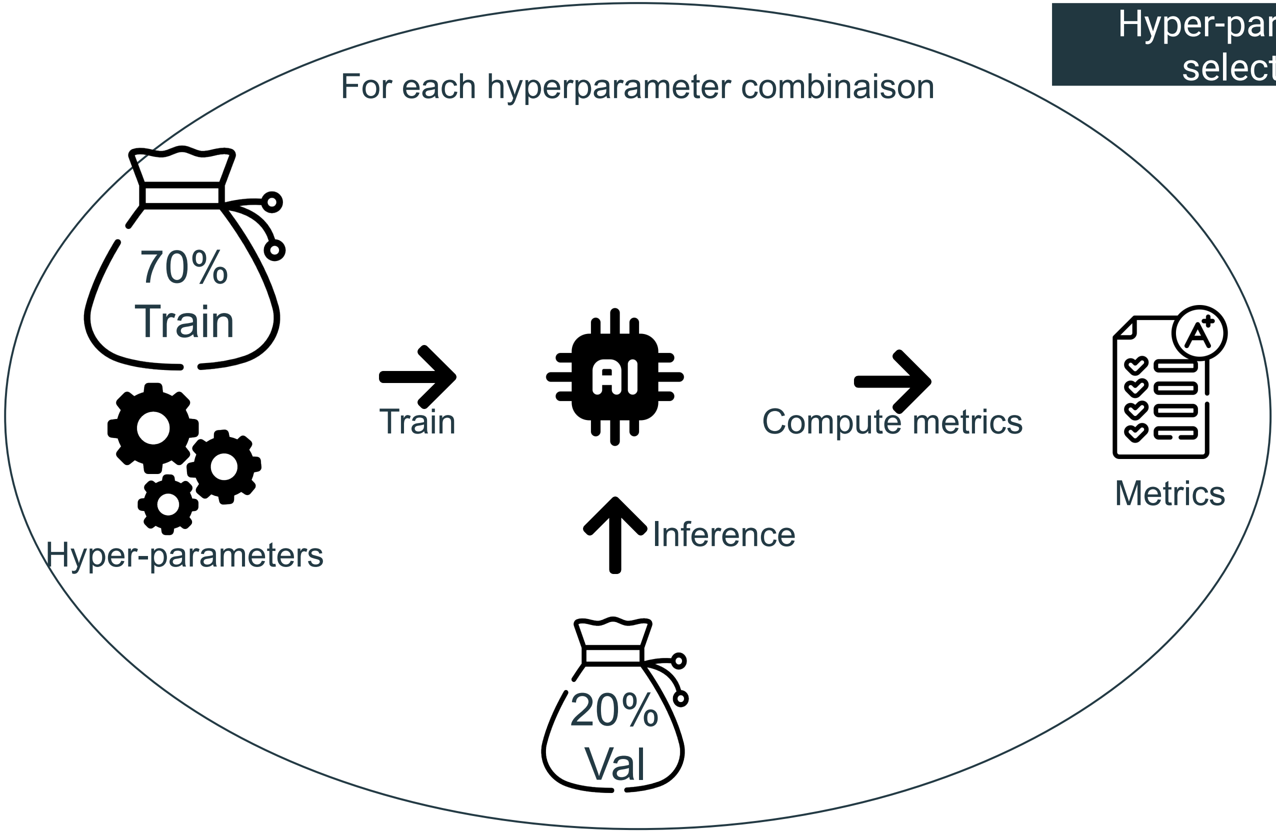
Hyper-parameter selection

Hyper-parameter selection | Retrain and performance

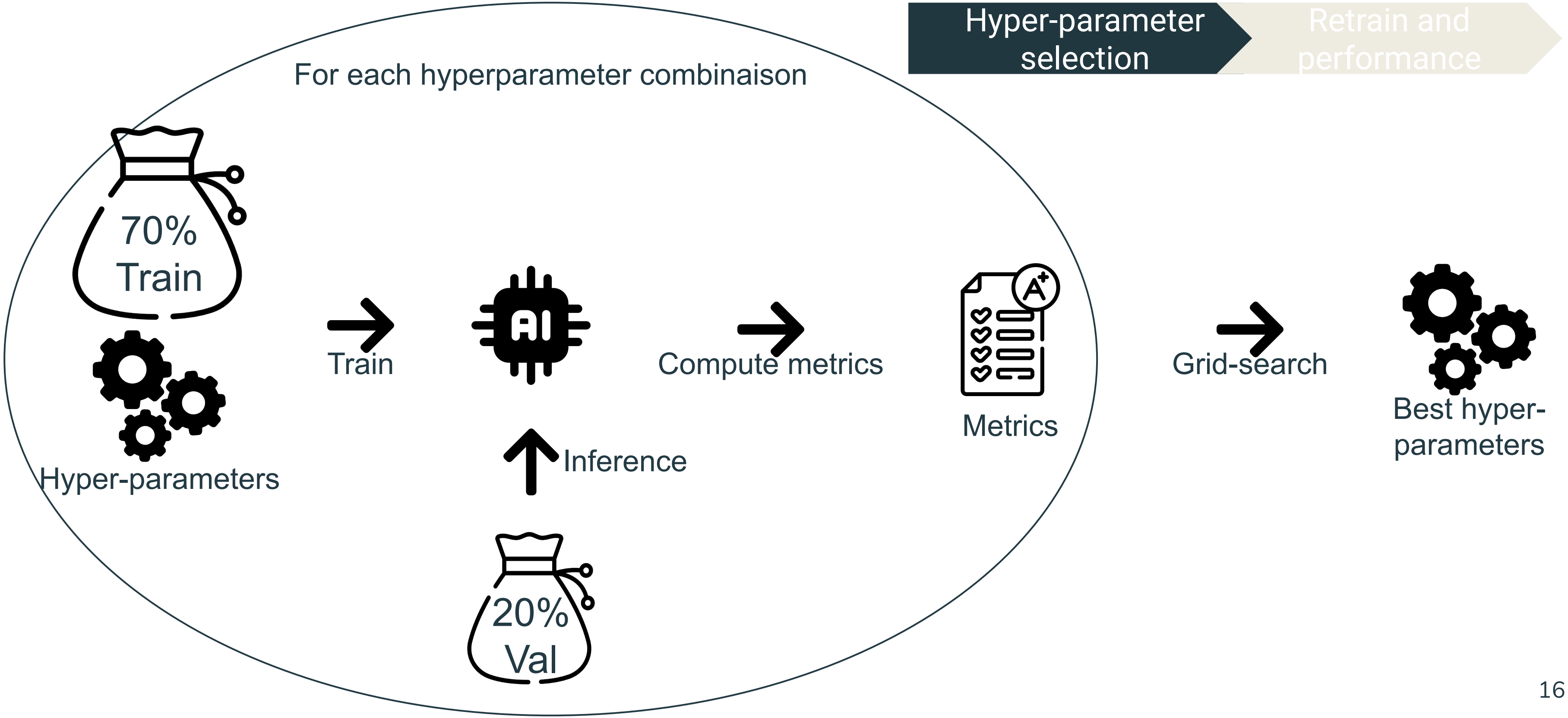


Hyper-parameter selection

Hyper-parameter selection | Retrain and performance

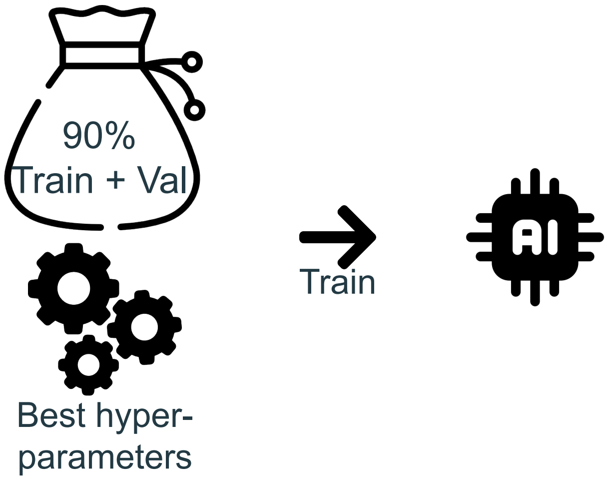


Hyper-parameter selection



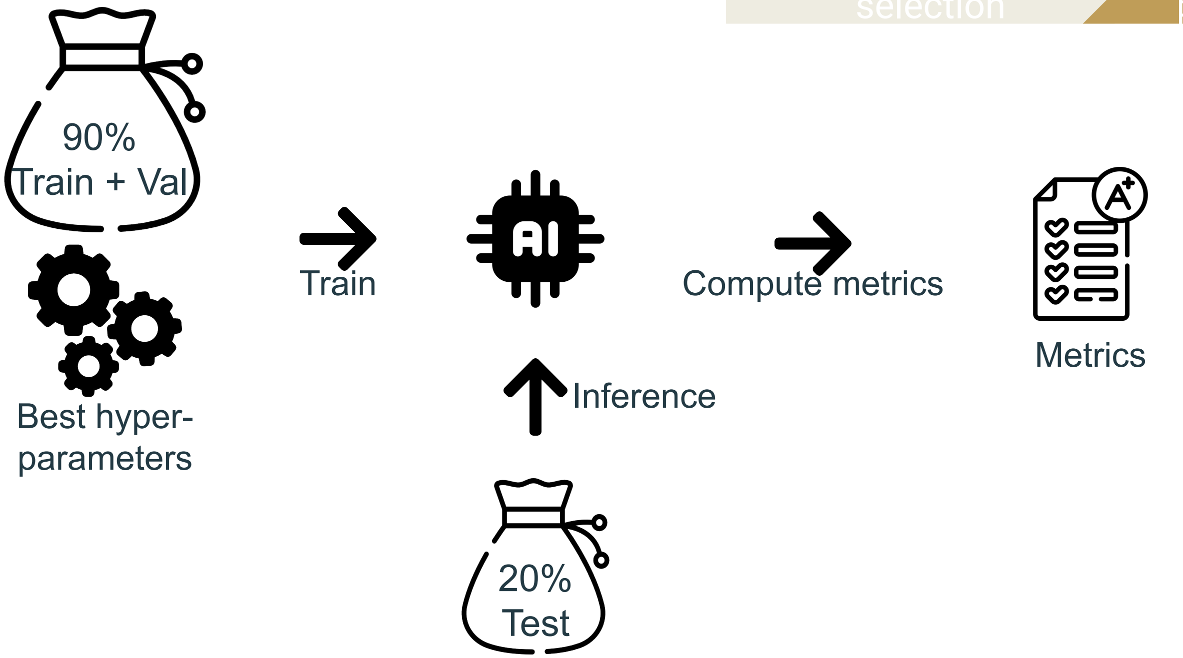
Retraining and performances computation

Hyper-parameter selection | Retrain and performance



Retraining and performances computation

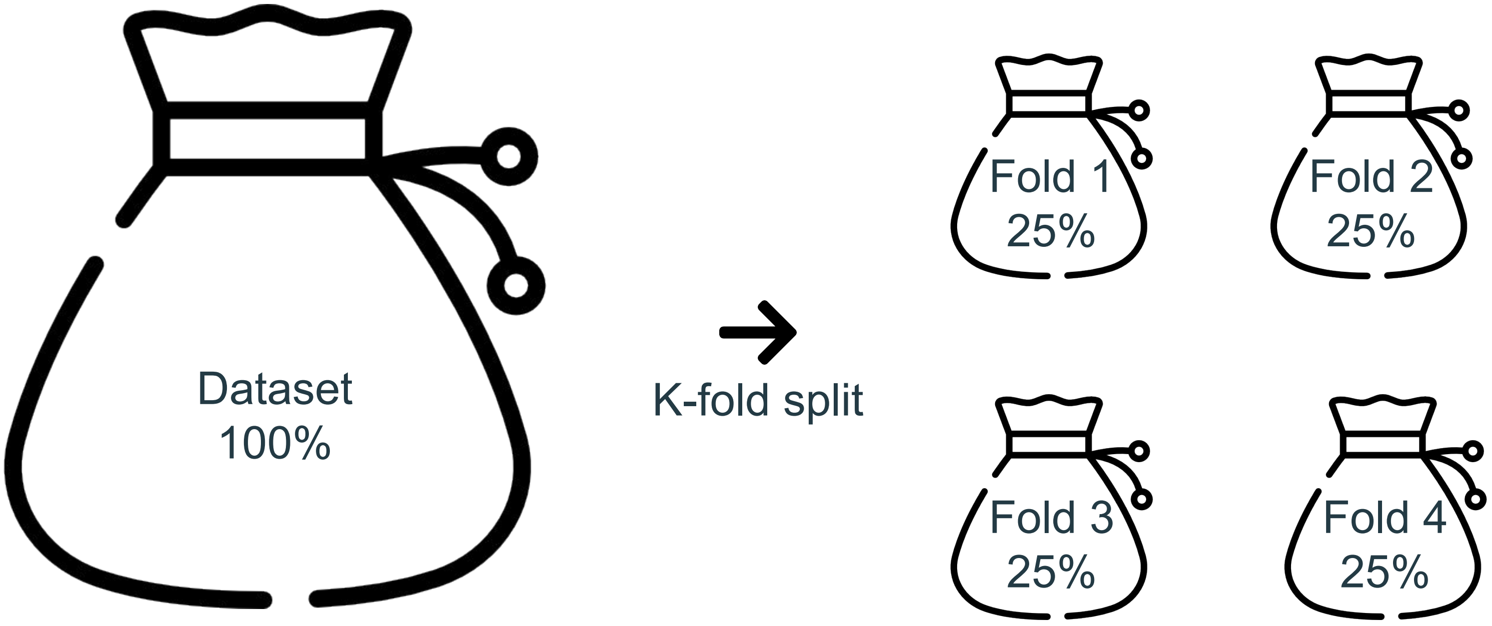
Hyper-parameter selection
Retrain and performance



Model comparison

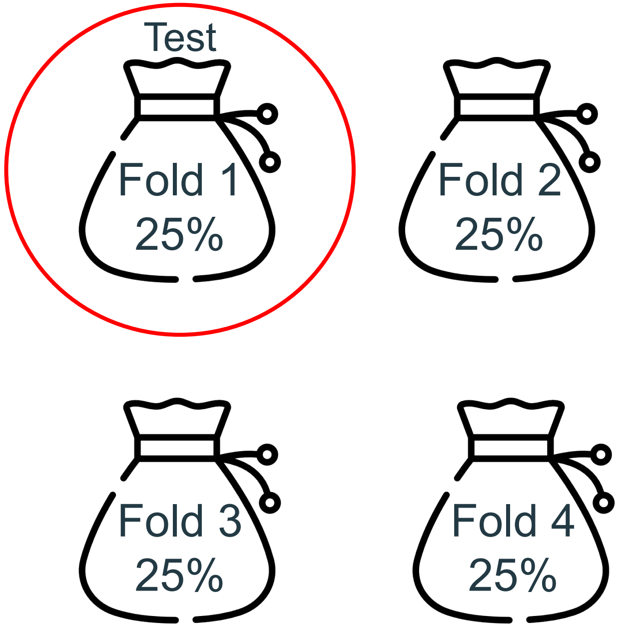
- One architecture → One performance measure
- No possible comparison between models
- We need several runs of the same experiment:
  - Statistics can be done on series of measures
  - This is called the **cross-validation**
  - We need to split our data into *folds*
  - This split is called the **k-fold**

K-fold split



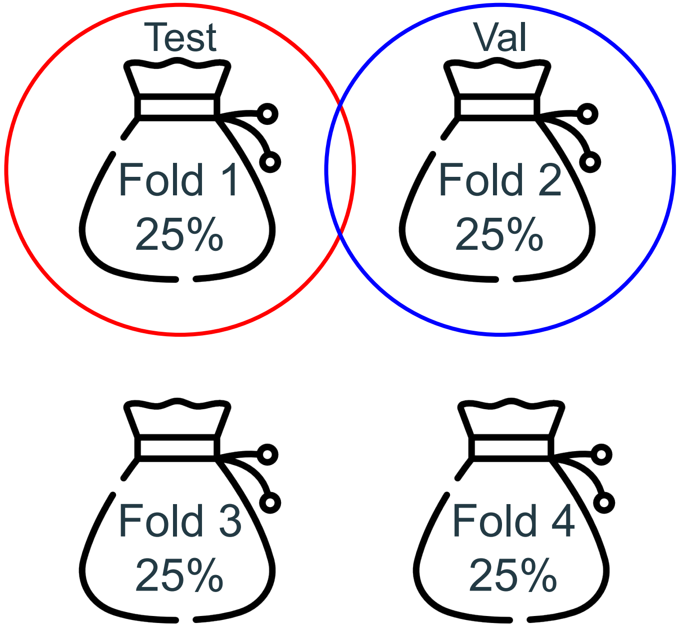
K-fold attribution

- For *i* from *1* to *k*
  - Fold *i* → test set
  - Fold *i+1* → validation set
  - Remaining folds → train set



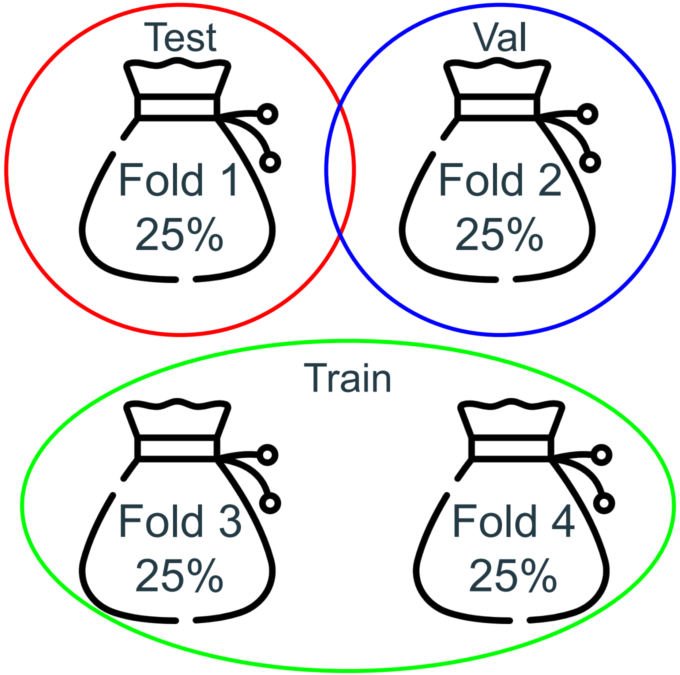
K-fold attribution

- For *i* from *1* to *k*
  - Fold *i* → test set
  - Fold *i+1* → validation set
  - Remaining folds → train set



K-fold attribution

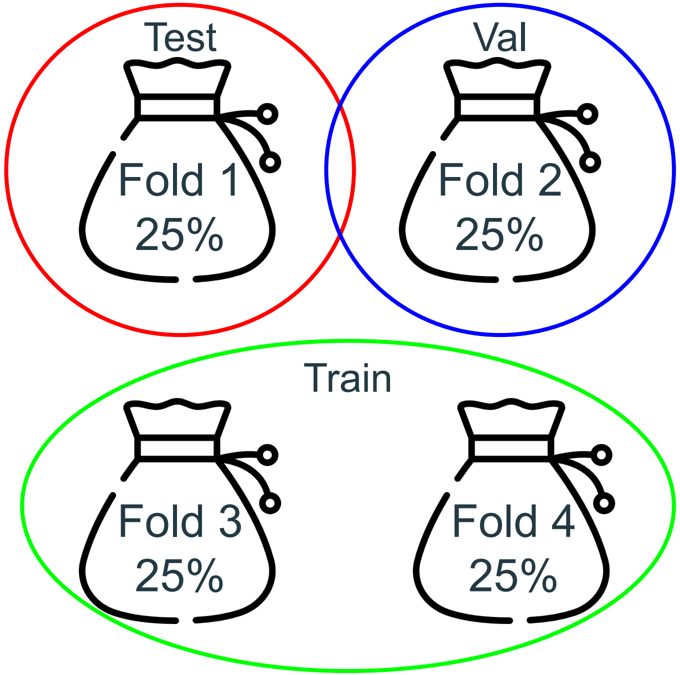
- For *i* from *1* to *k*
  - Fold *i* → test set
  - Fold *i+1* → validation set
  - Remaining folds → train set



Cross-validation

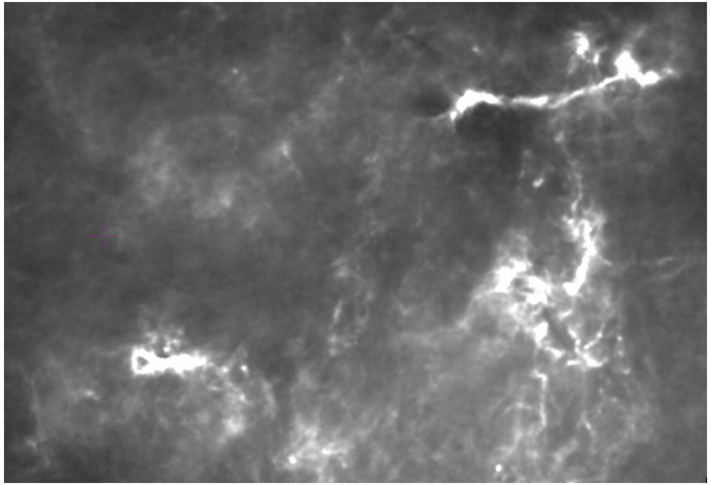
- Perform the usual training procedure with the corresponding sets
- We train *k* models for *k* performances
- Every sample is seen exactly once in test

⚠ Computation time multiplicated by *k*



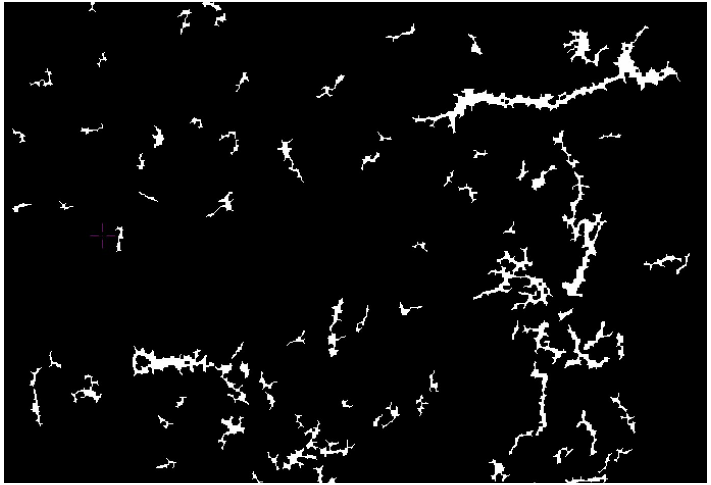
Semantic segmentation task

- Classification task
- Pixel level
- Two classes: Filament - Background



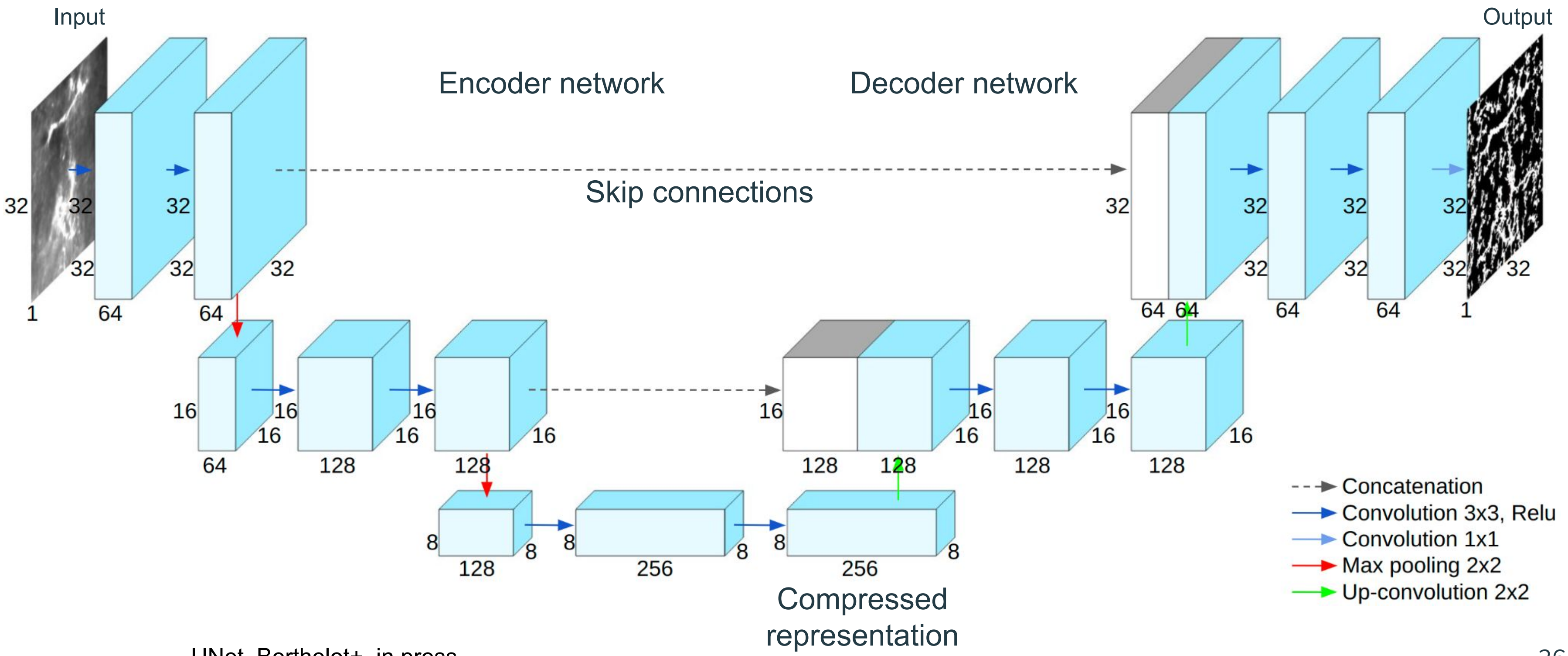
H_2 column density: Hi-GAL dataset, Molinari+ 2010

Semantic segmentation



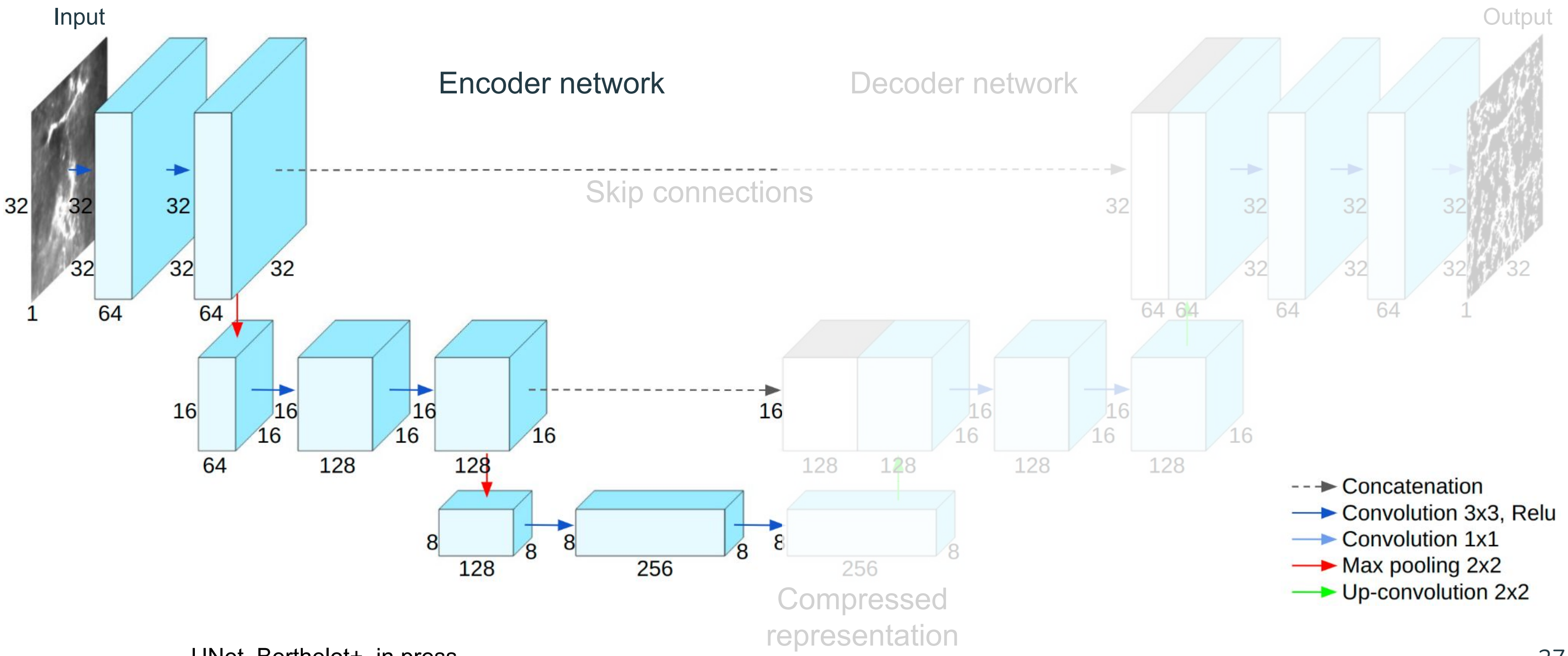
Schisano+2020

UNet models: Auto-encoder



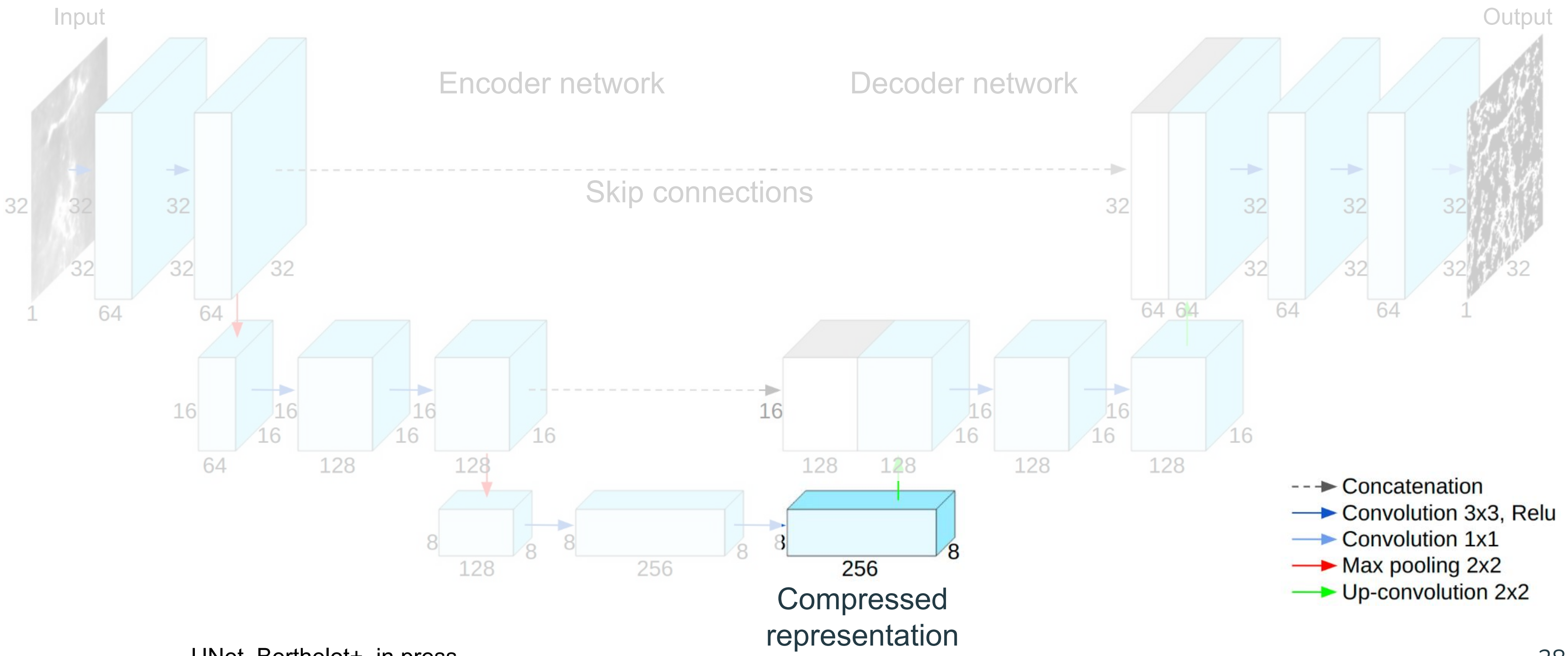
UNet, Berthelot+, in press

UNet models: Auto-encoder



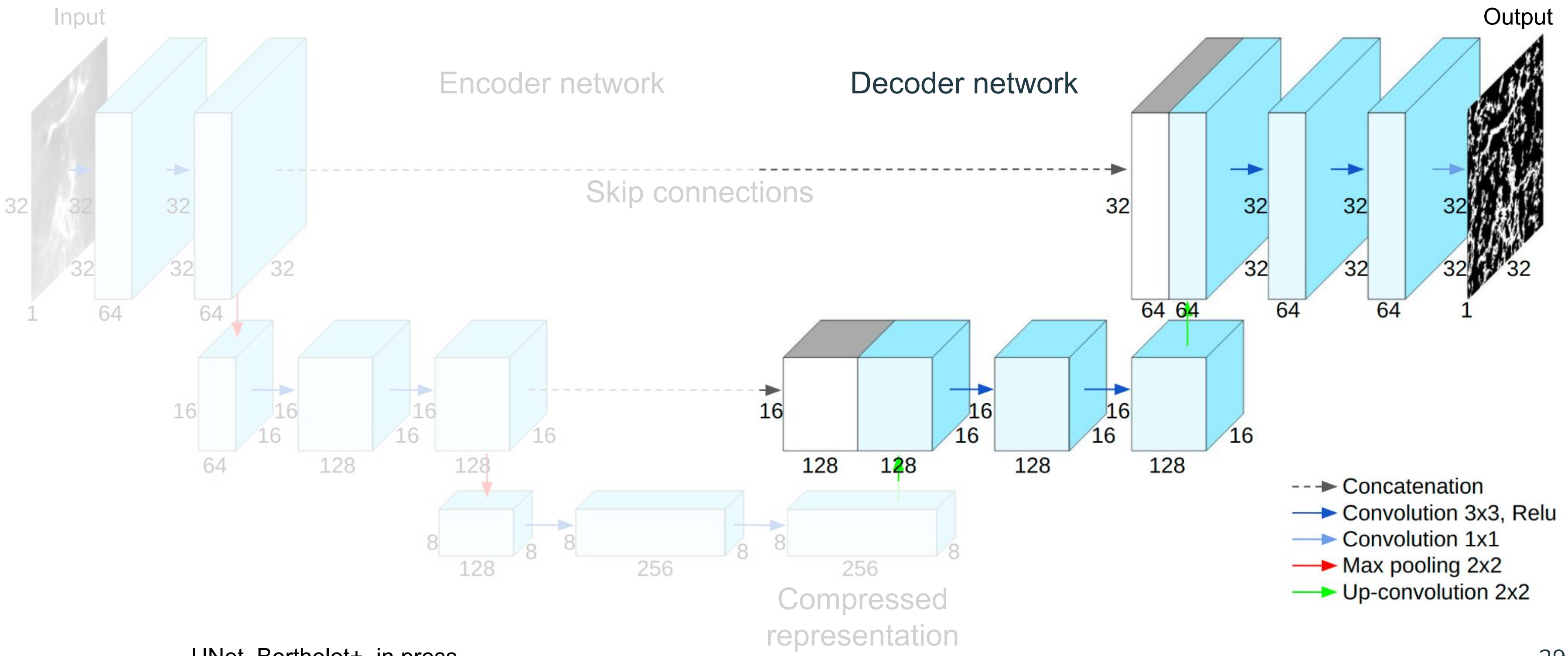
UNet, Berthelot+, in press

UNet models: Auto-encoder



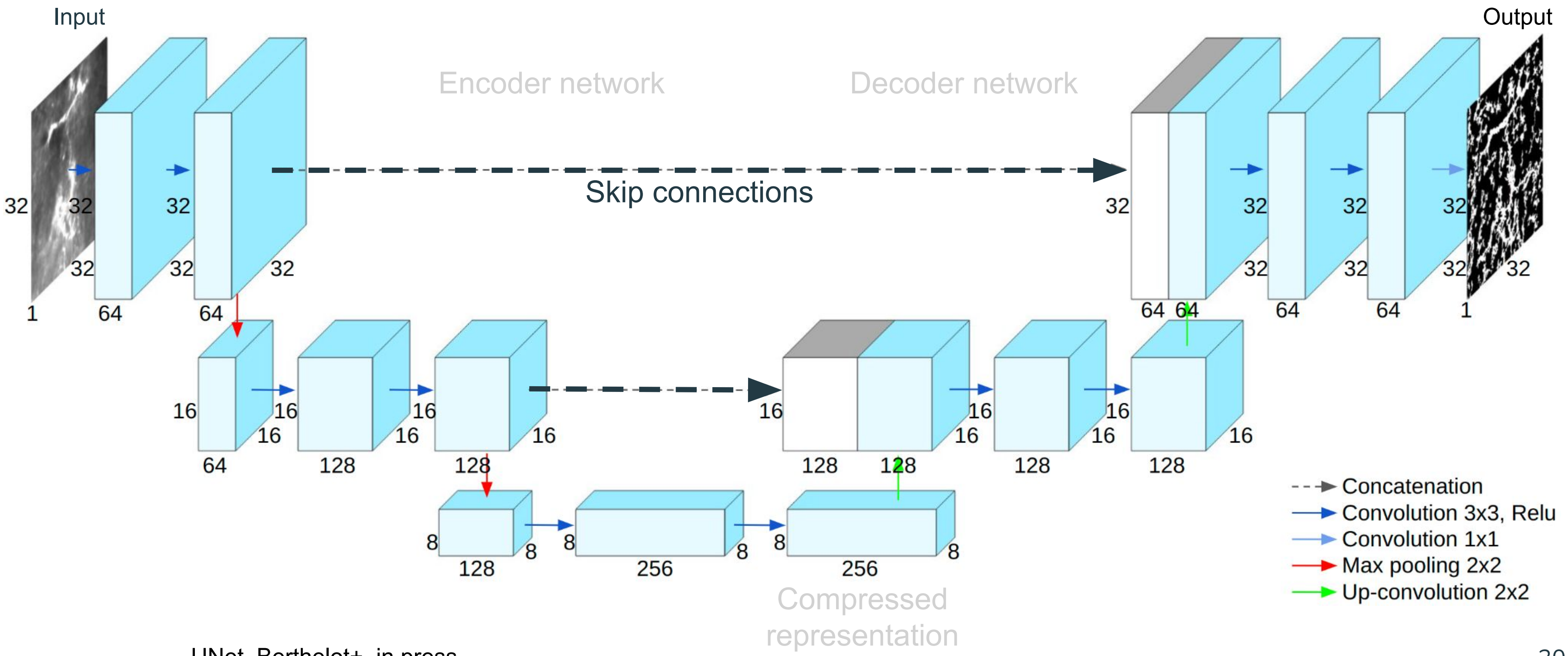
UNet, Berthelot+, in press

UNet models: Auto-encoder



UNet, Berthelot+, in press

UNet models: Auto-encoder



UNet, Berthelot+, in press

Dataset: Hi-GAL

Hi-GAL: Description

- Herschel: far-IR space telescope (2009-2014)
- Survey of the Galactic plane in five photometric bands with the PACS and SPIRE instruments (Molinari+2010)
- Column density map $N(H_2)$ of the whole Gp derived (Elia+2013, Schisano+2020)

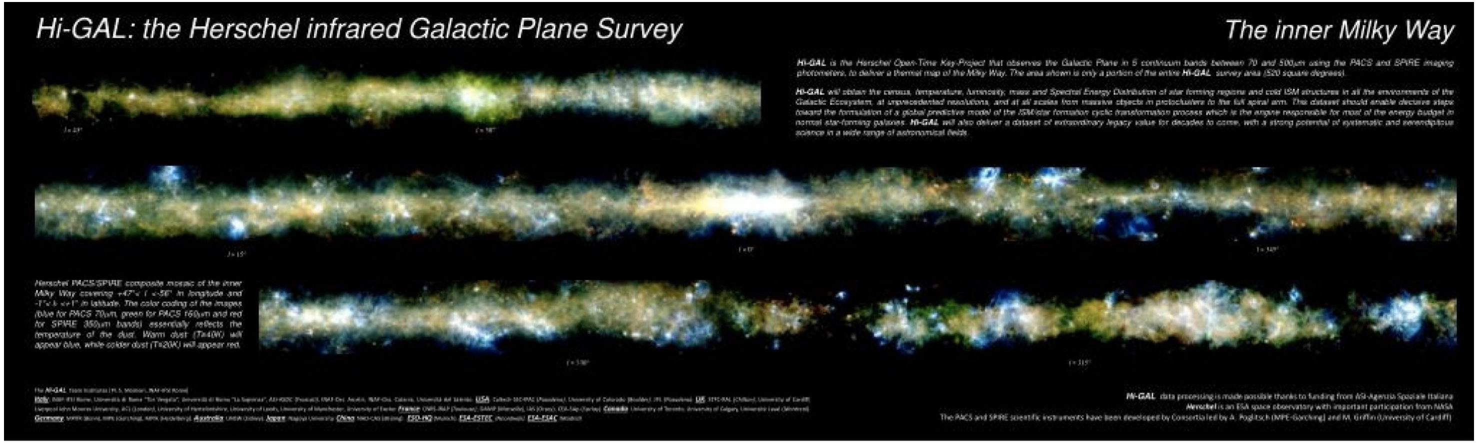
- Image of size: 1500 x 110000 pixels
- Pixel value: 4×10^{20} - 4×10^{23} H_2 molecules cm^{-2}
- **32069 filaments** extracted in the Gp (Schisano+2020) → Labels

Inference and learning on one single image

Hi-GAL: One single image

- We can not perform a direct k-fold split

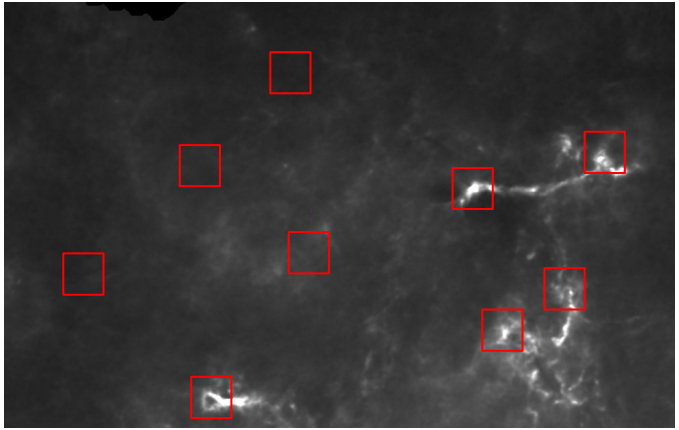
Split our image into smaller ones (32x32 pixels patches)



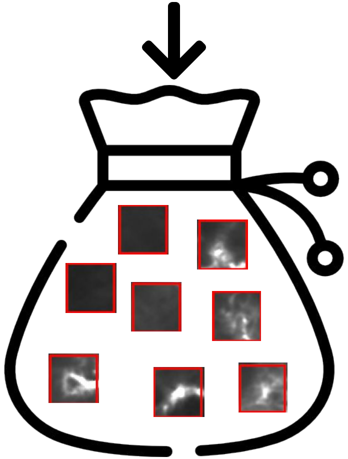
Herschel Hi-GAL survey of the inner Galactic plane (Molinari+2010)

Patch-based learning

- One patch = one sample
  - It allows us to do k-fold split
  - Every patch appears exactly once in test
- We can reconstruct the Gp map from the k-fold

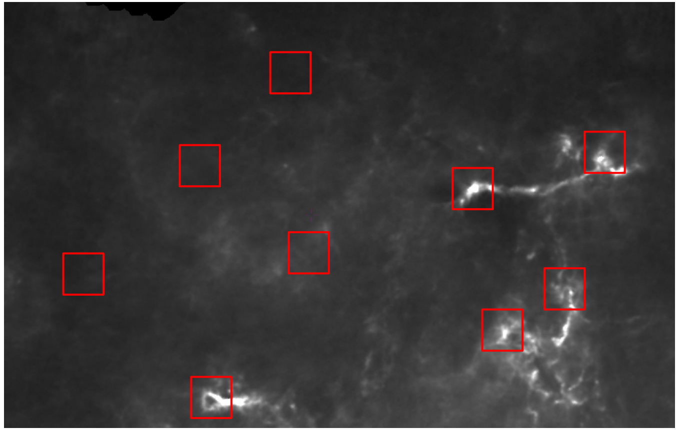


H_2 column density: Hi-GAL dataset, Molinari+ 2010

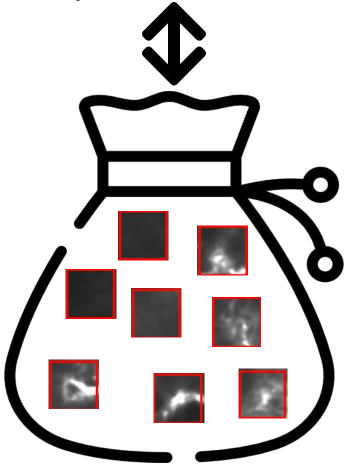


Patch-based learning

- One patch = one sample
  - It allows us to do k-fold split
  - Every patch appears exactly once in test
- We can reconstruct the Gp map from the k-fold



H_2 column density: Hi-GAL dataset, Molinari+ 2010

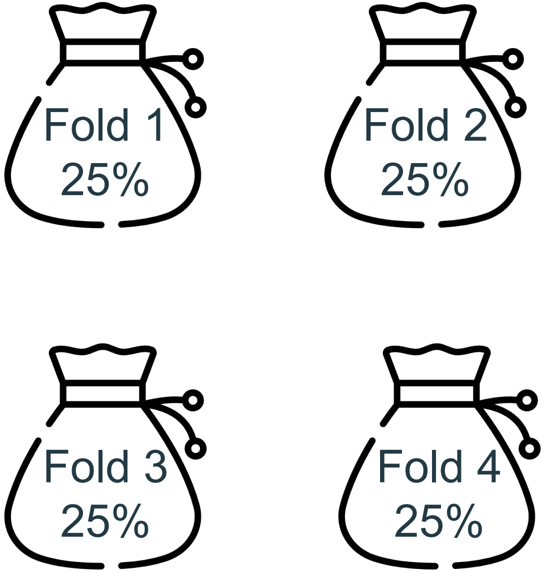


Random k-fold strategy

Randomly assign each sample to a fold

- Imbalanced positions might imply imbalanced folds:
    - Position imbalance
    - Filament properties
    - Class balance

Pseudo random k-fold

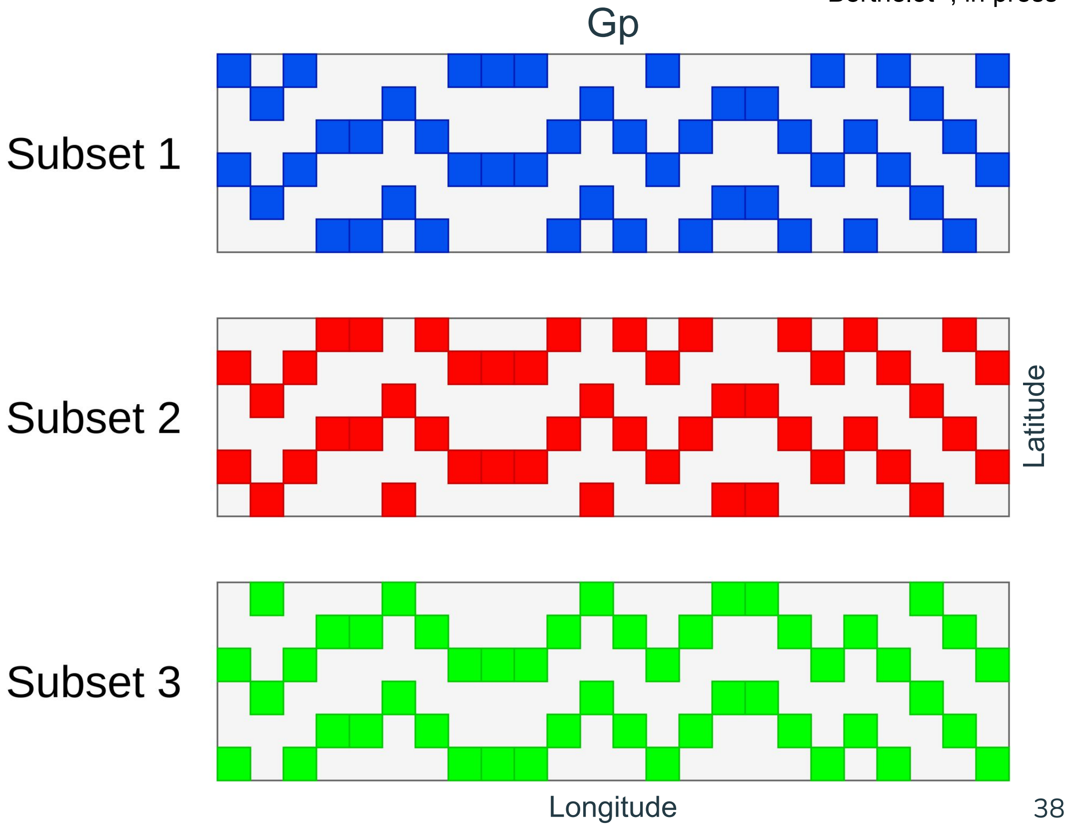


Pseudo random k-fold strategy

Pseudo random k-fold,
Berthelot+, in press

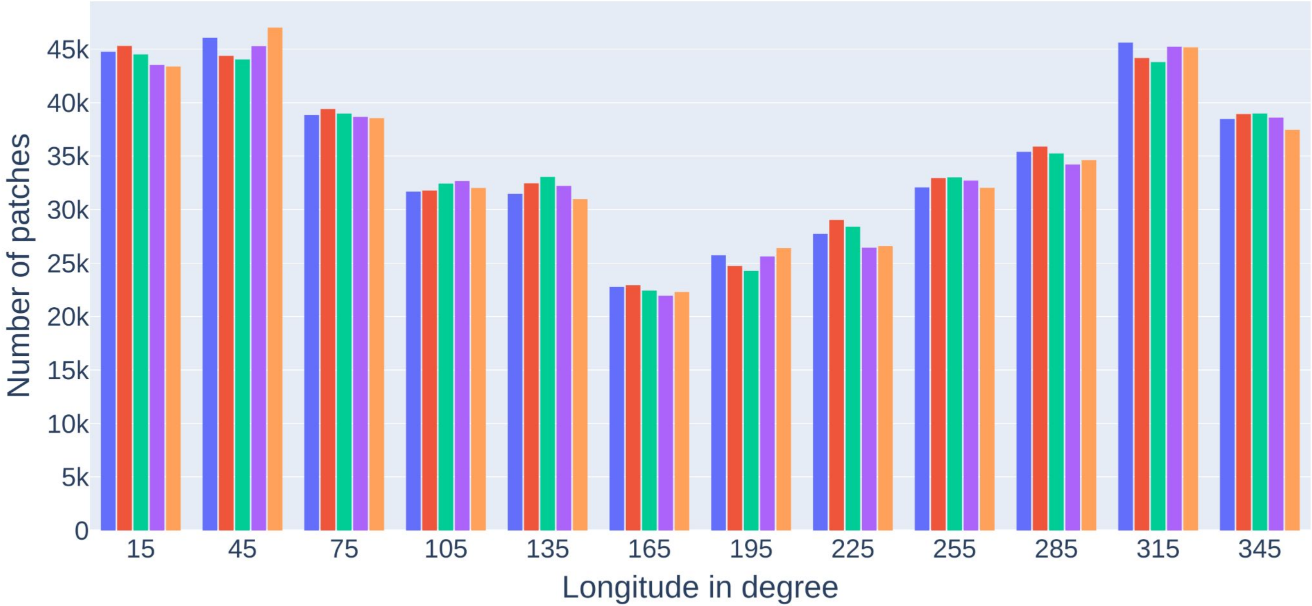
For a given longitude

- Assign the first patch to the *ith* fold (*i* random)
- Assign the second patch to the *i+1th* fold
- Assign the third patch to the *i+2th* fold



Position distribution with pseudo random k-folds

For each position, the number of patch should be the same in each fold



One colour → One fold

Position distribution, Berthelot+, in press

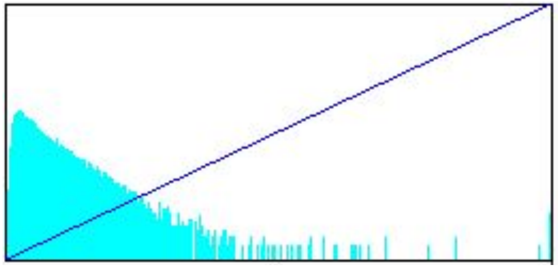
Normalization

Challenges of normalization

- Normalization is mandatory in ML
  - Data consistency
  - Mitigate outlier's impact

N(H_2) large range

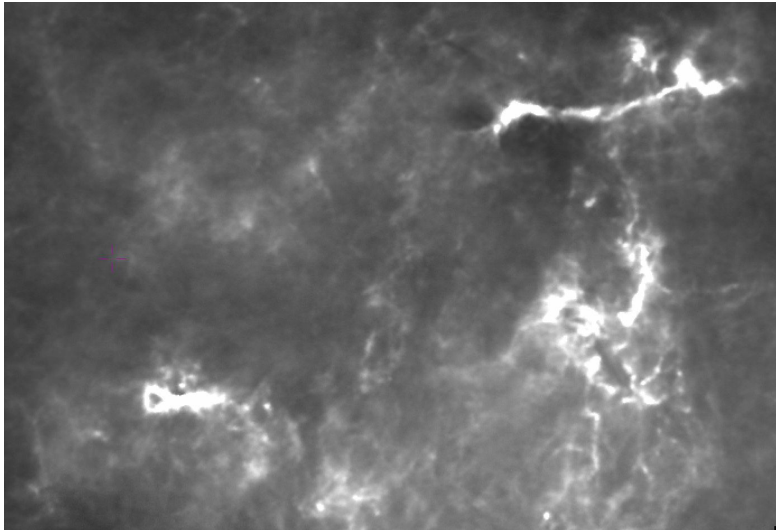
Global normalisation will prevent low density filament detection



4×10^{20} - 4×10^{23} H_2 molecules cm^{-2}

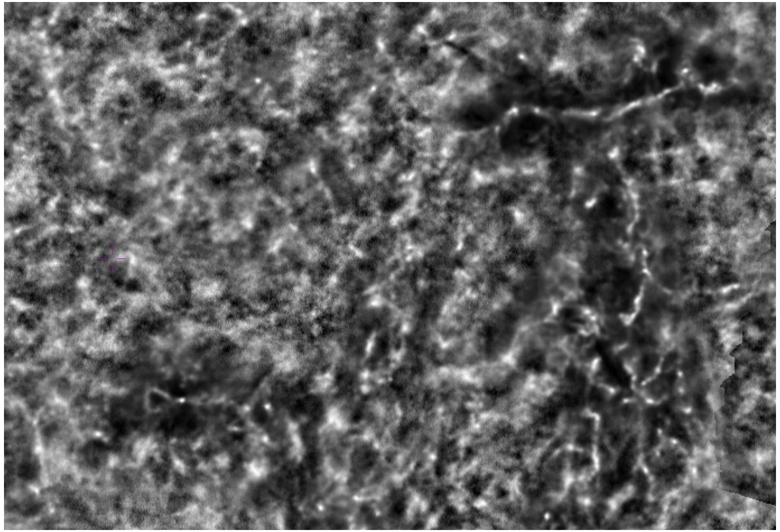
Local min-max normalisation (patch-based)

Local min-max map: *revealing the structure*



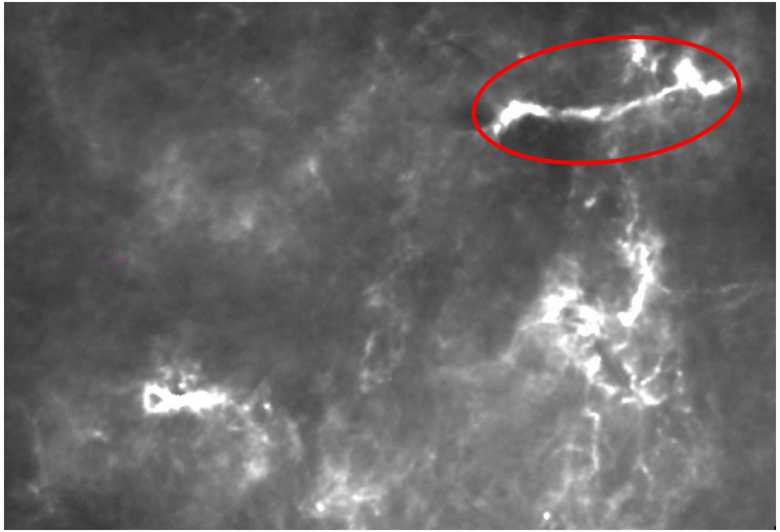
H_2 column density: Hi-GAL dataset, Molinari+ 2010

Local normalisation



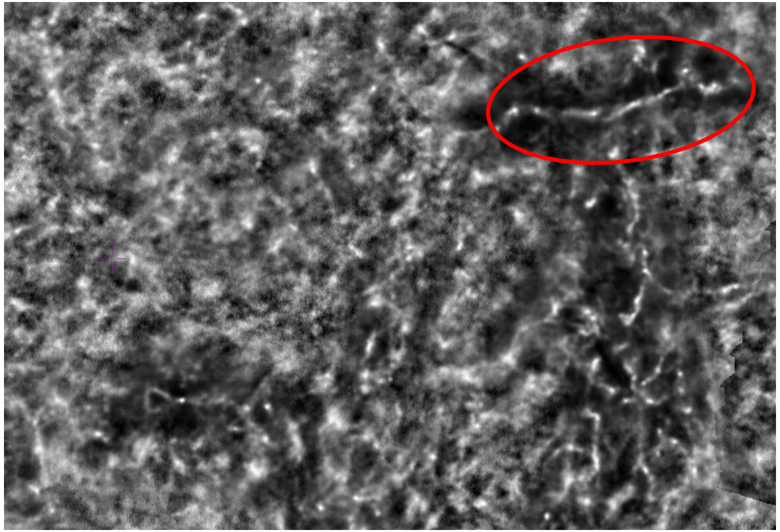
Local min-max normalization, Berthelot+ in press

Local min-max map: *revealing the structure*



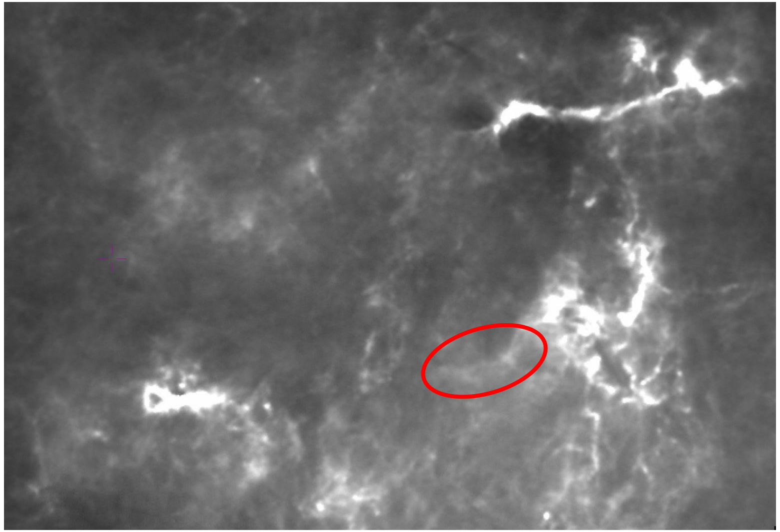
H_2 column density: Hi-GAL dataset, Molinari+ 2010

→ Local normalisation



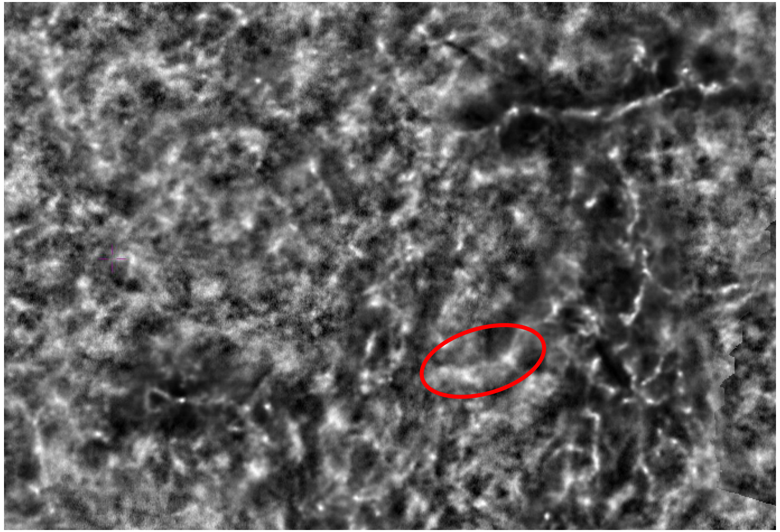
Local min-max normalization, Berthelot+ in press

Local min-max map: *revealing the structure*



H_2 column density: Hi-GAL dataset, Molinari+ 2010

Local normalisation



Local min-max normalization, Berthelot+ in press

Semi-supervised learning

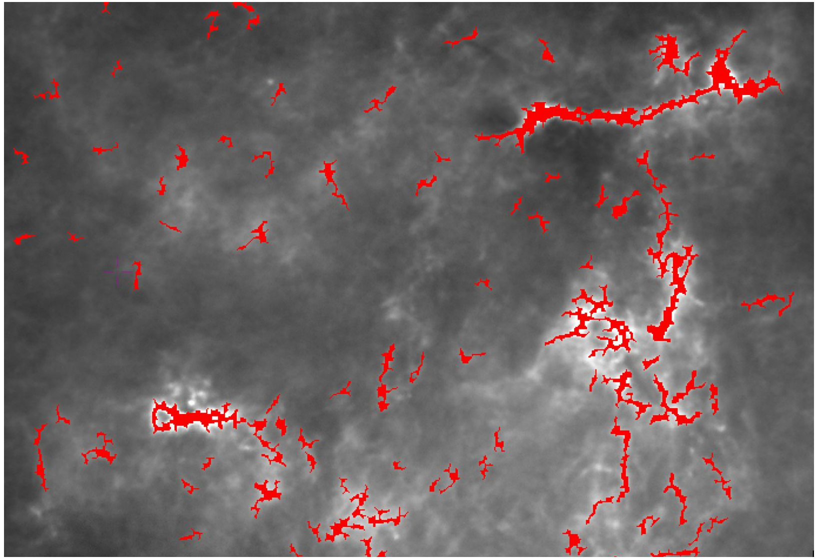
Using the usual supervised learning paradigm

Reproduce the current results:

- Reduce the computation time
- Fusion of the processing steps

⚠ Extraction of only known filaments

→ We want to detect new filaments



H_2 column density: Hi-GAL dataset, Uncomplete annotation

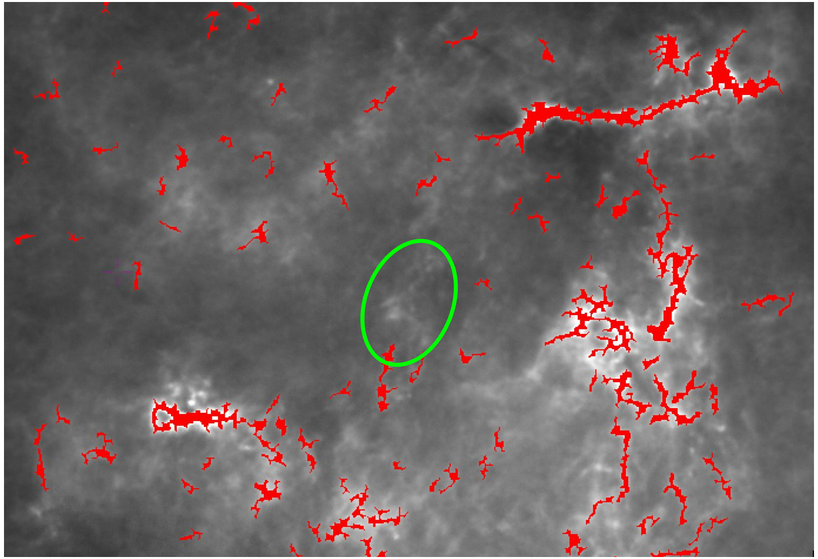
Using the usual supervised learning paradigm

Reproduce the current results:

- Reduce the computation time
- Fusion of the processing steps

⚠ Extraction of only known filaments

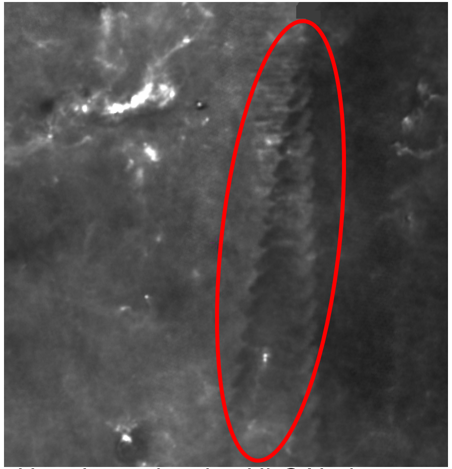
→ We want to detect new filaments



H_2 column density: Hi-GAL dataset, Uncomplete annotation

Using noisy inputs

- Artefacts



H_2 column density: Hi-GAL dataset, artefacts

- Missing input

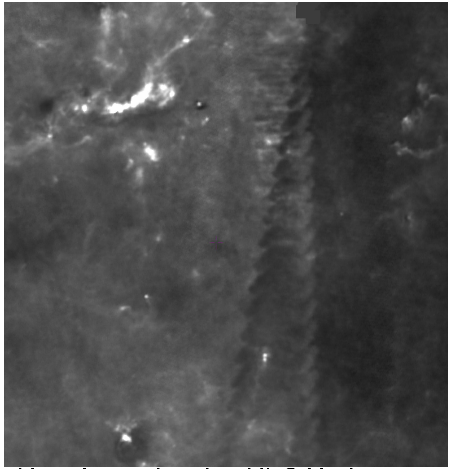
H_2 column density: Hi-GAL dataset, missing values

→ We want to avoid learning and metric computation on noisy pixels

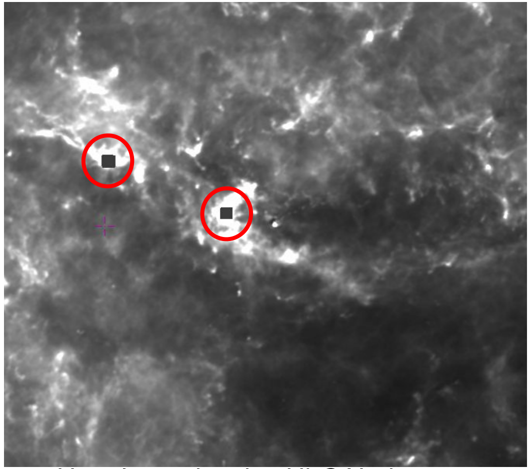
Using noisy inputs

- Artefacts

- Missing input



H_2 column density: Hi-GAL dataset, artefacts

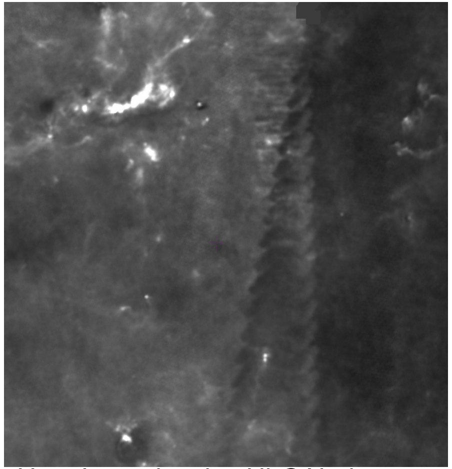


H_2 column density: Hi-GAL dataset, missing values

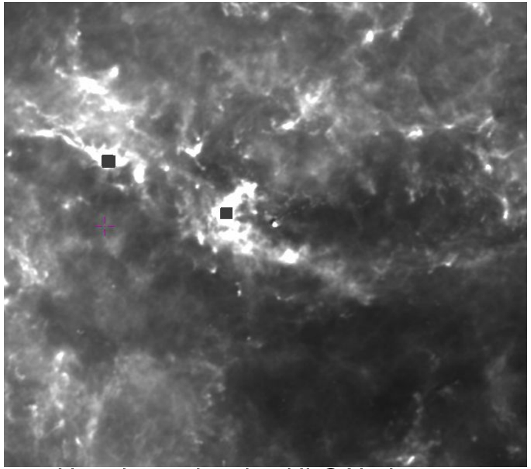
→ We want to avoid learning and metric computation on noisy pixels

Using noisy inputs

- Artefacts
- Missing input



H_2 column density: Hi-GAL dataset, artefacts



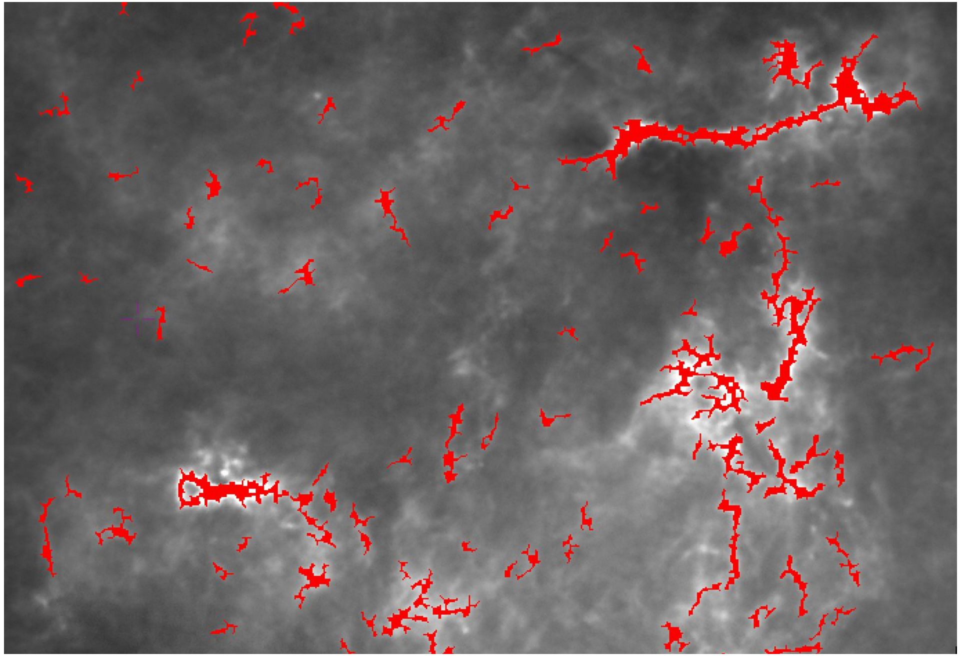
H_2 column density: Hi-GAL dataset, missing values

➔ We want to avoid learning and metric computation on noisy pixels

A simple semi-supervised learning strategy

- Filament candidate from Schisano+2020
- Background pixels are pixels with a lower column density than a handcrafted local threshold
- Rest is unknown

⚠ Training and metric computation is done only on *known* pixels

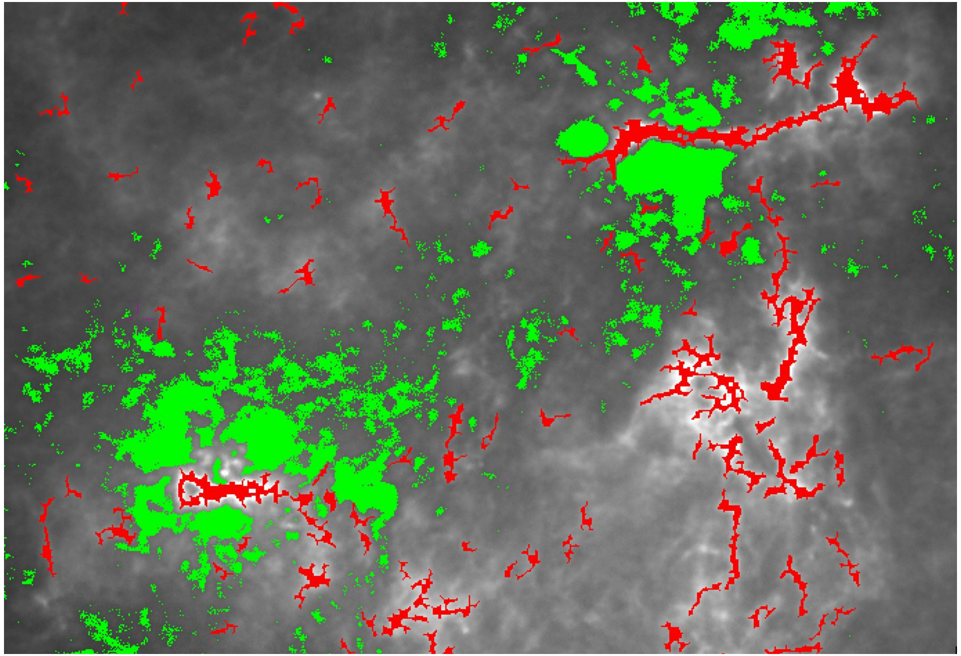


H_2 column density: Hi-GAL dataset, Pixel labels, background pixels are in green and filament pixels are in red. Rest is unknown.

A simple semi-supervised learning strategy

- Filament candidate from Schisano+2020
- Background pixels are pixels with a lower column density than a handcrafted local threshold
- Rest is unknown

Training and metric computation is done only on *known* pixels

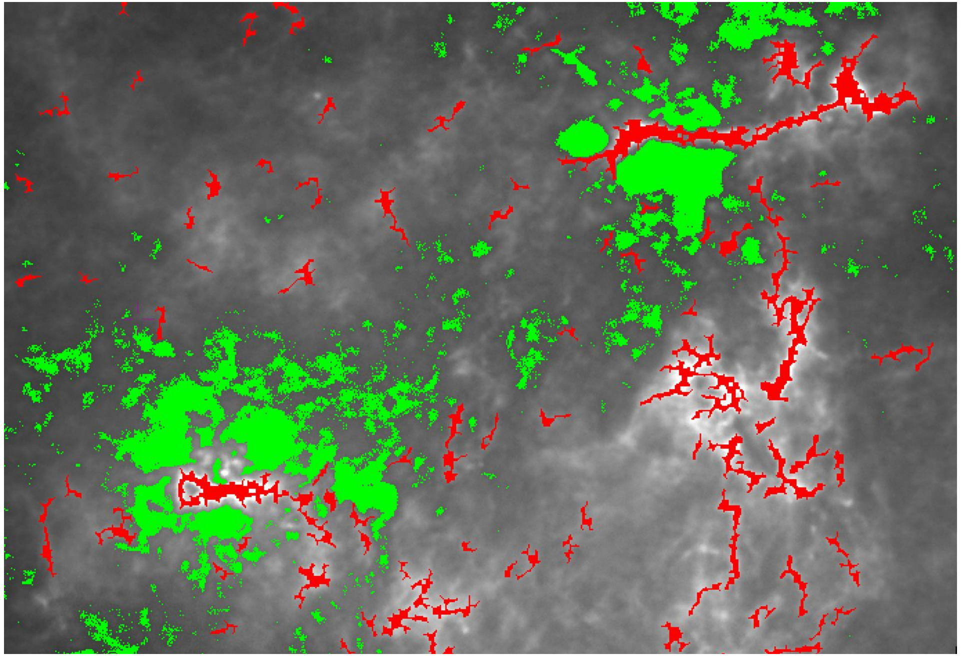


H_2 column density: Hi-GAL dataset, Pixel labels, background pixels are in green and filament pixels are in red. Rest is unknown.

A simple semi-supervised learning strategy

- Filament candidate from Schisano+2020
- Background pixels are pixels with a lower column density than a handcrafted local threshold
- Rest is unknown

⚠ Training and metric computation is done only on *known* pixels (red & green)



H_2 column density: Hi-GAL dataset, Pixel labels, background pixels are in green and filament pixels are in red. Rest is unknown.

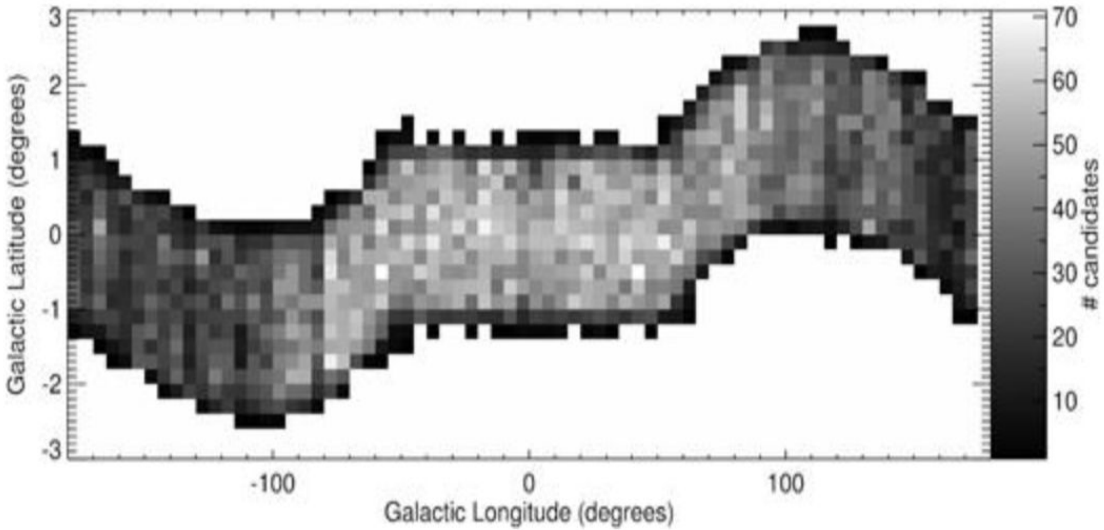
PE-UNets: Adding physics information to improve performance

Filament-position relation

- Filament properties:
  - Shape
  - Contrast ratio
  - Orientation
  - Column density
  - Length
  - Width

- Number of filaments

→ Position might be an important information for filament detection



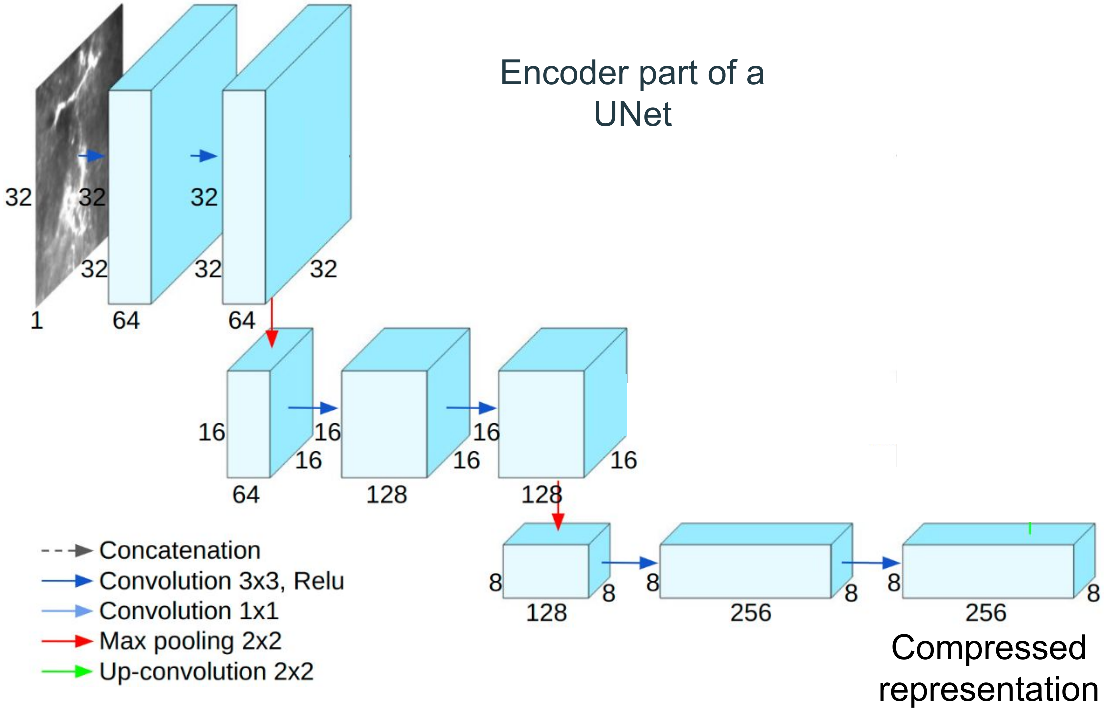
Filament distribution along the Galactic plane, Schisano+2020

Is the position information present in the column density?

→ Can we train a model to predict the position of a patch given as input?

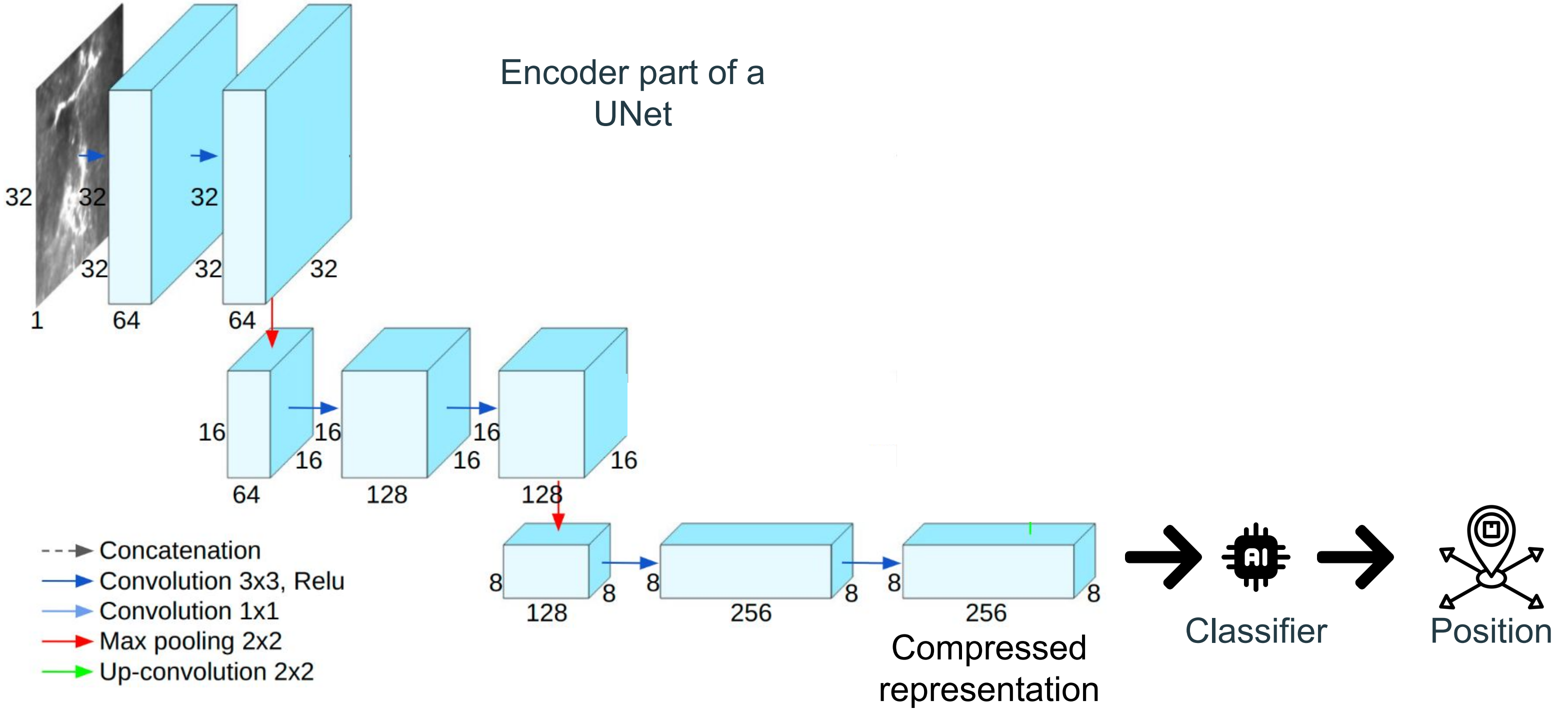
Is the position information present in the column density?

→ Can we train a model to predict the position of a patch given as input?



Is the position information present in the column density?

→ Can we train a model to predict the position of a patch given as input?



Can we train a model to predict the position of a patch given as input?

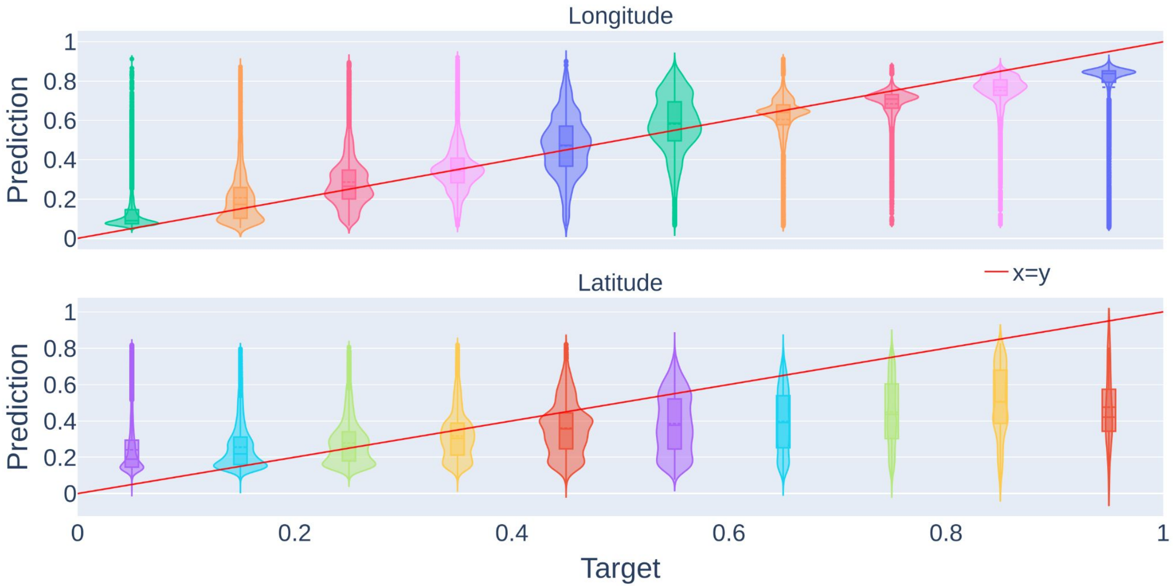
Classifier:

- Linear(4096, 2048)
- Relu()
- Dropout(0.5)
- Linear(2048, 1024)
- Relu()
- Dropout(0.5)
- Linear(1024, 2)

Can we train a model to predict the position of a patch given as input?

Classifier:

- Linear(4096, 2048)
- Relu()
- Dropout(0.5)
- Linear(2048, 1024)
- Relu()
- Dropout(0.5)
- Linear(1024, 2)



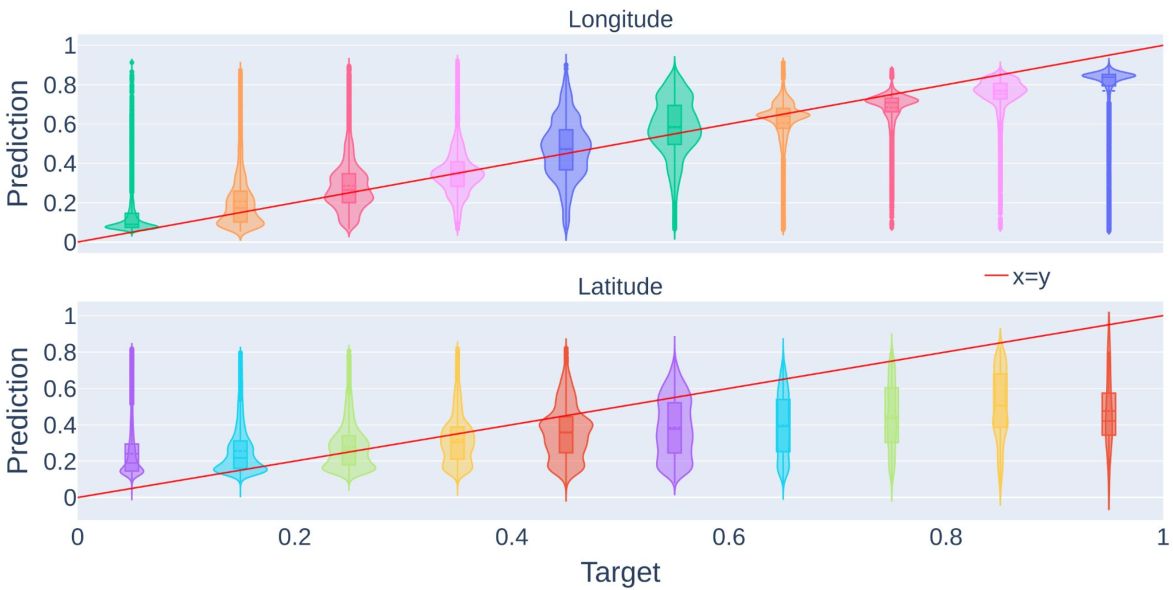
Predicted position against true position, Berthelot+, in press

Can we train a model to predict the position of a patch given as input?

Classifier:

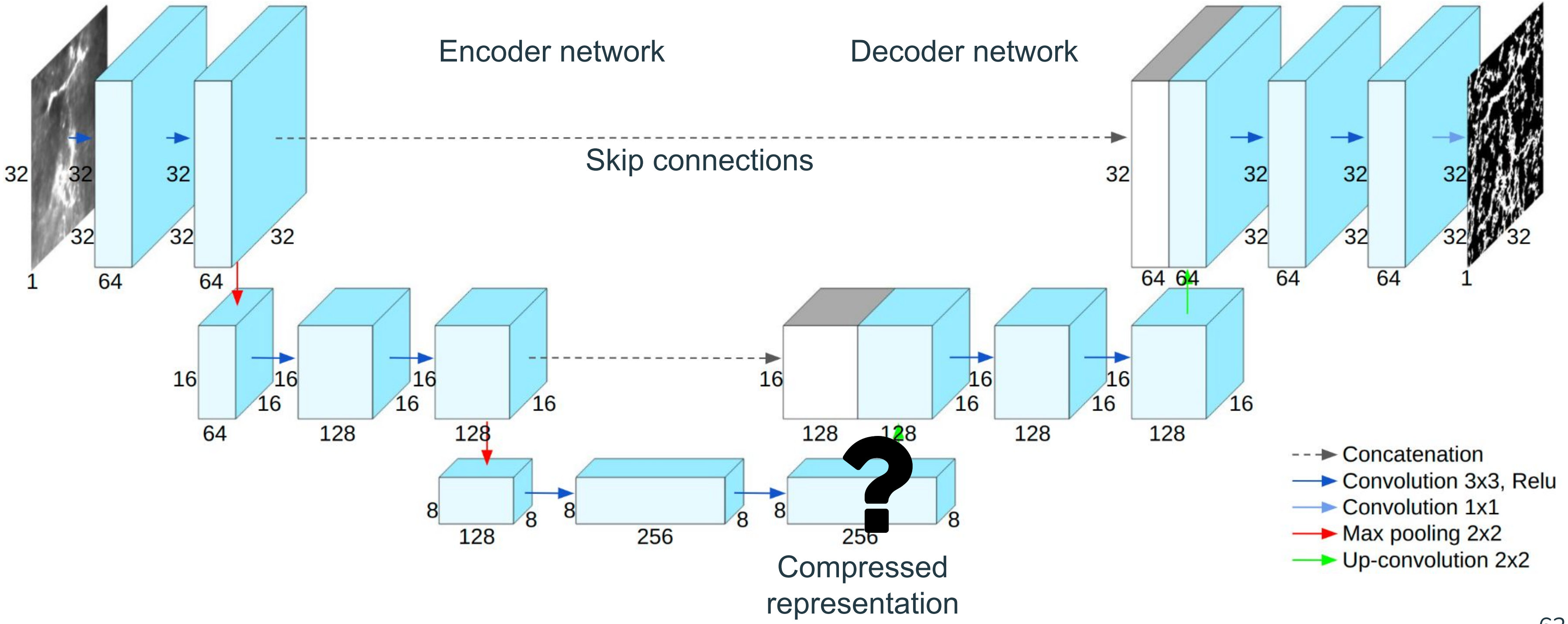
- Linear(4096, 2048)
- Relu()
- Dropout(0.5)
- Linear(2048, 1024)
- Relu()
- Dropout(0.5)
- Linear(1024, 2)

→ To some extent, it is possible to predict the position from the density



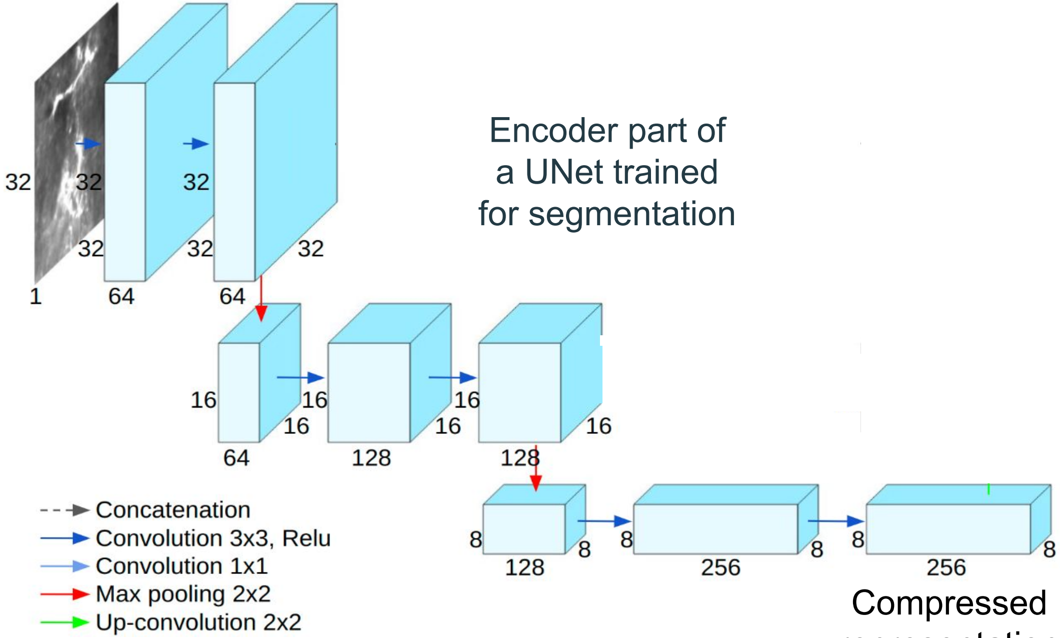
Predicted position against true position, Berthelot+, in press

Is the position present in the compressed representation?



Is the position present in the compressed representation?

→ Can we train a classifier to predict the position of a patch given a compressed representation of a patch?

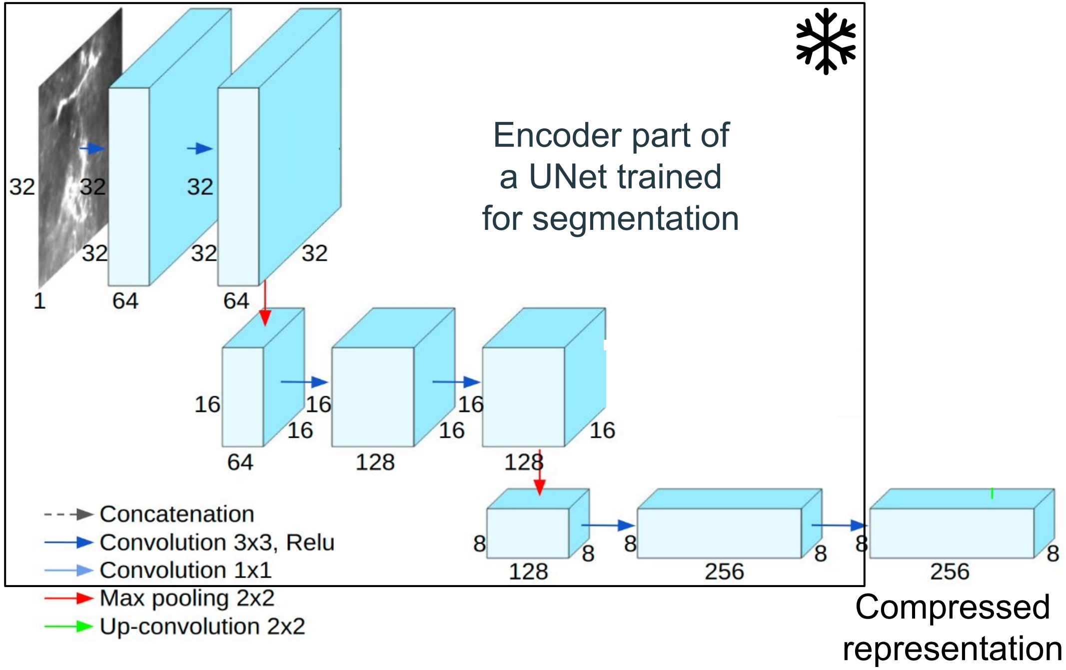


Encoder part of a UNet trained for segmentation

Compressed representation

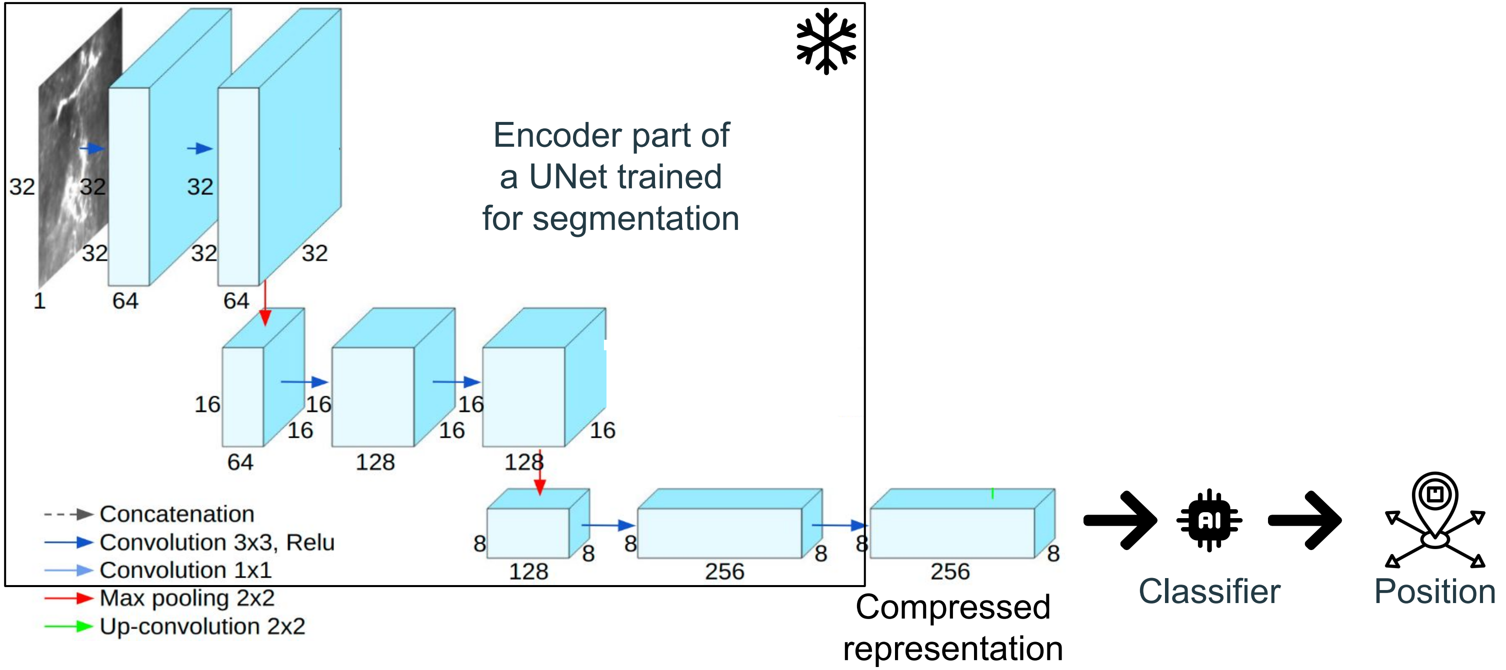
Is the position present in the compressed representation?

→ Can we train a classifier to predict the position of a patch given a compressed representation of a patch?



Is the position present in the compressed representation?

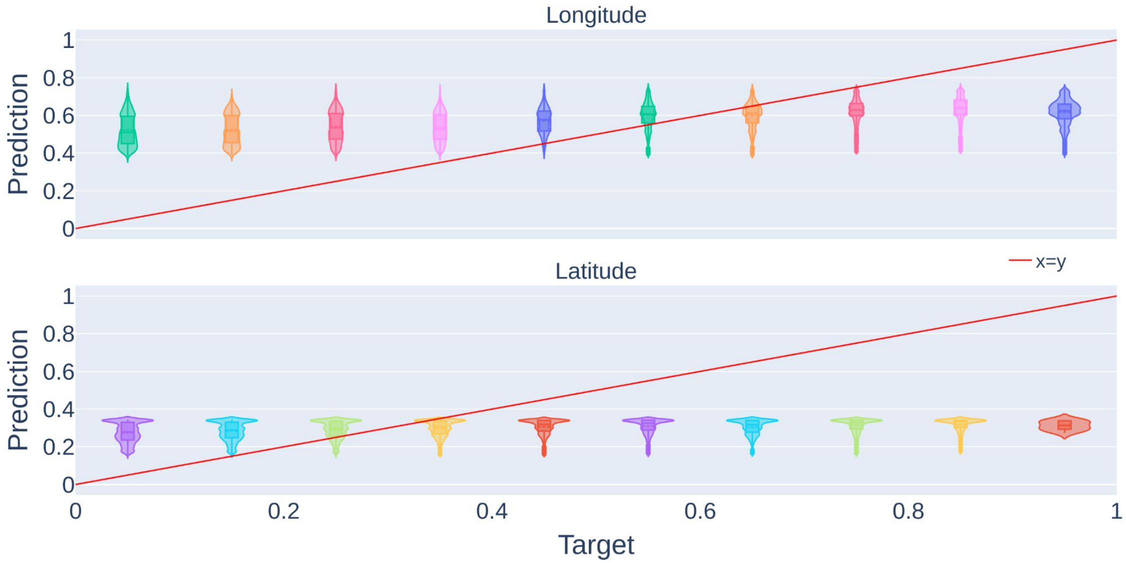
→ Can we train a classifier to predict the position of a patch given a compressed representation of a patch?



Can we train a classifier to predict the position of a patch given a compressed representation of a patch?

Classifier:

- Linear(4096, 2048)
- Relu()
- Dropout(0.5)
- Linear(2048, 1024)
- Relu()
- Dropout(0.5)
- Linear(1024, 2)



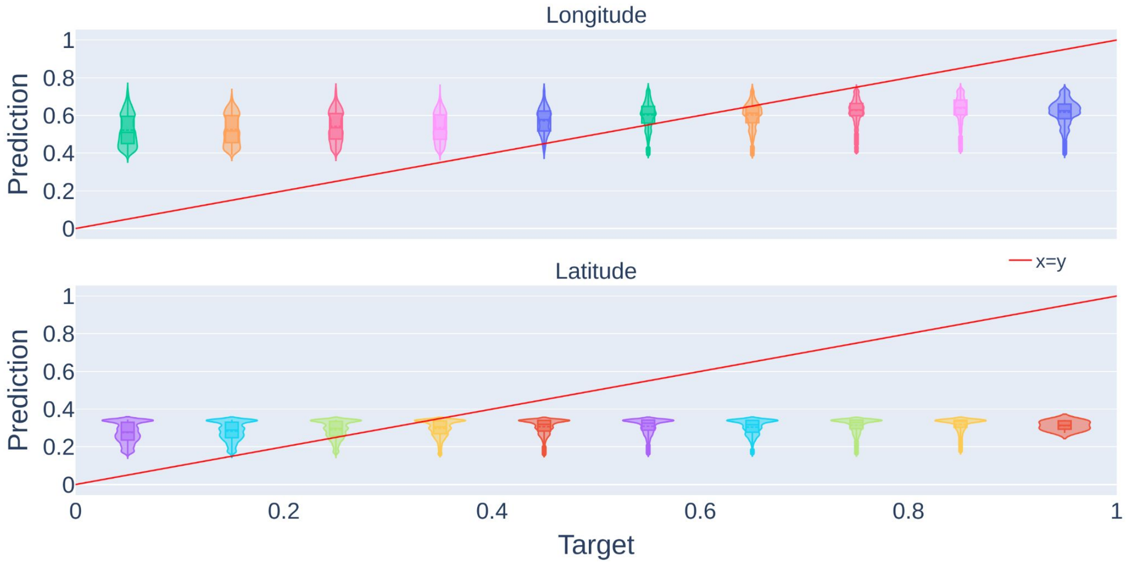
Predicted position against true position, Berthelot+, in press

Can we train a classifier to predict the position of a patch given a compressed representation of a patch?

Classifier:

- Linear(4096, 2048)
- Relu()
- Dropout(0.5)
- Linear(2048, 1024)
- Relu()
- Dropout(0.5)
- Linear(1024, 2)

→ UNet trained for segmentation do not use the position information contained in the patches

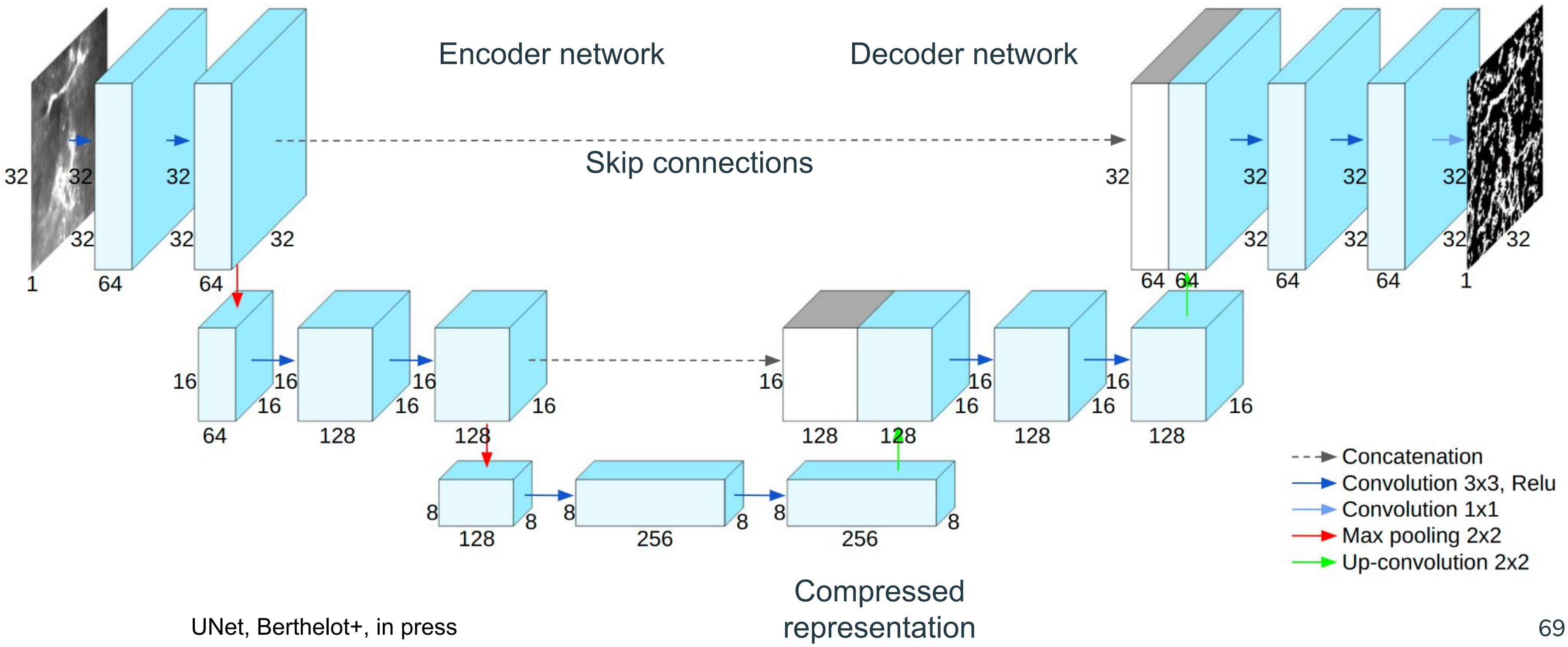


Predicted position against true position, Berthelot+, in press

How to improve UNet performances?

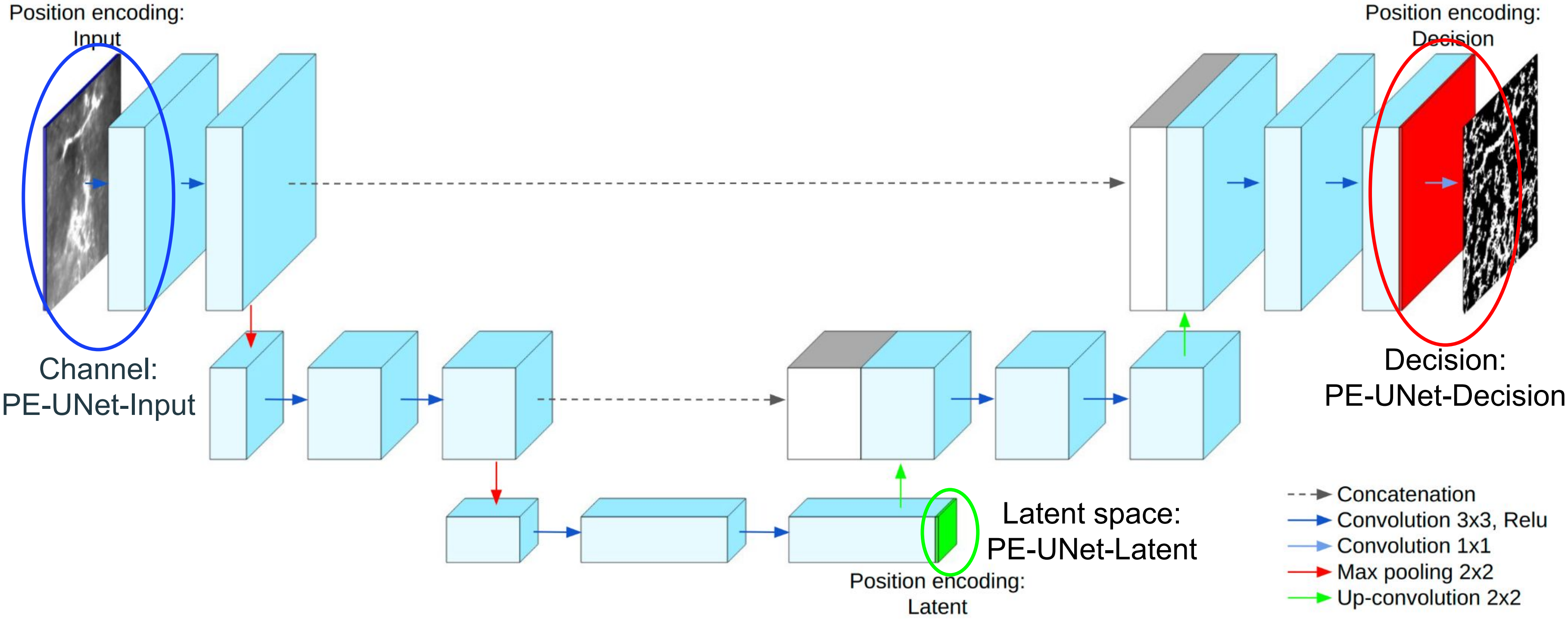
- UNet models don’t use position information for segmentation
- Position of filament in the Gp might be important (physical conditions)
- Let’s give position as input to the UNet models

UNet



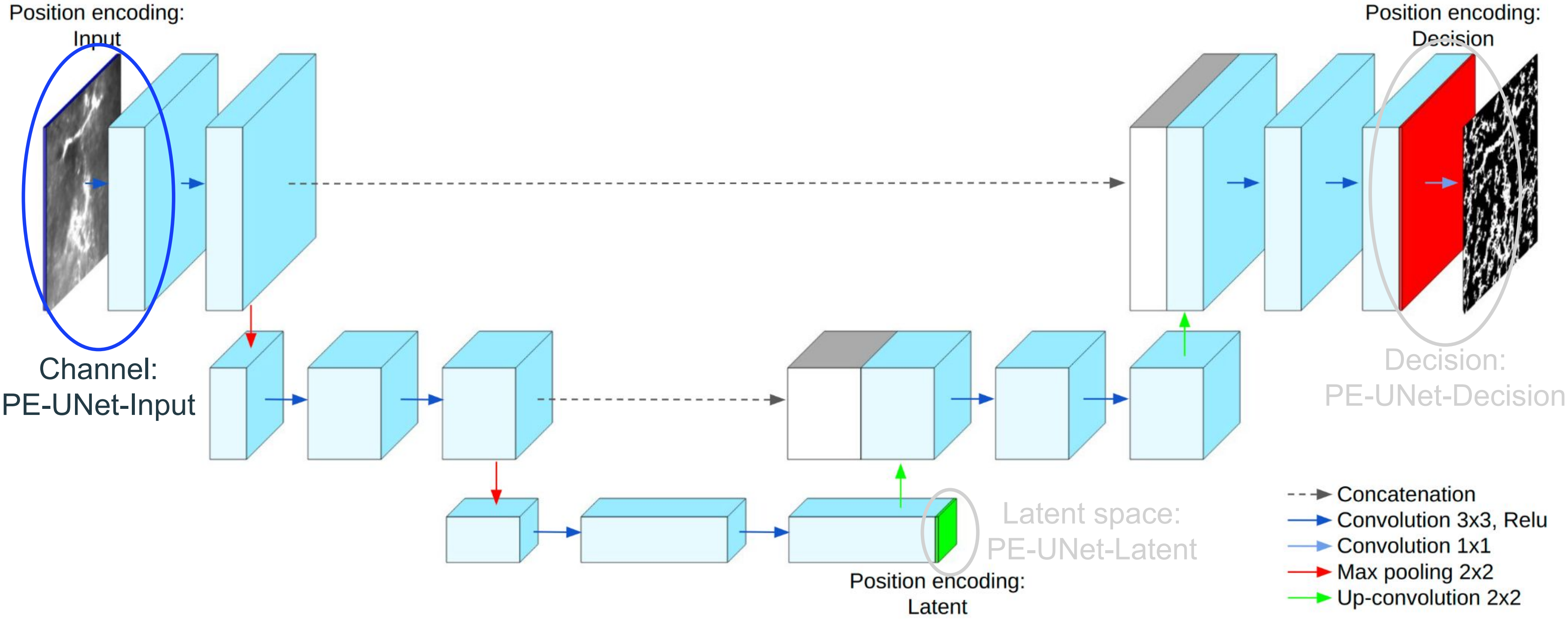
UNet, Berthelot+, in press

PE-UNets



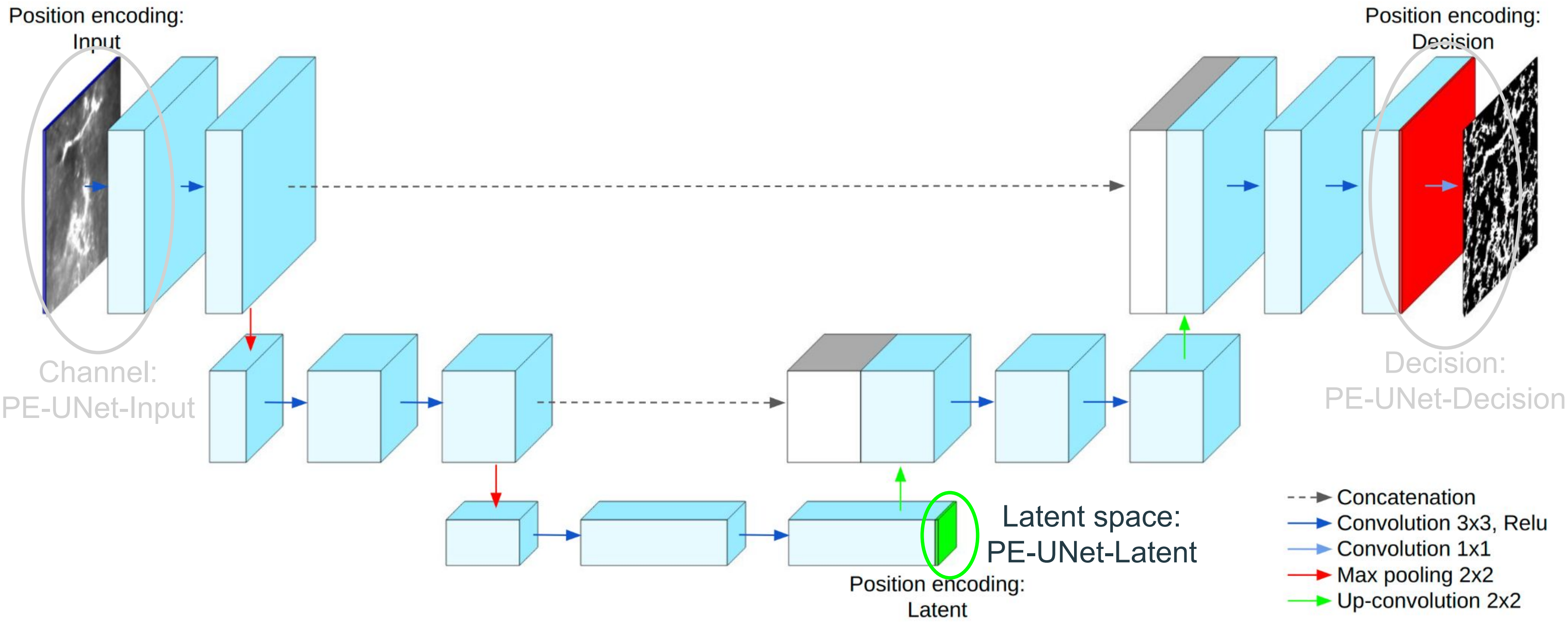
PE-UNets, Berthelot+, in press

PE-UNet-Input



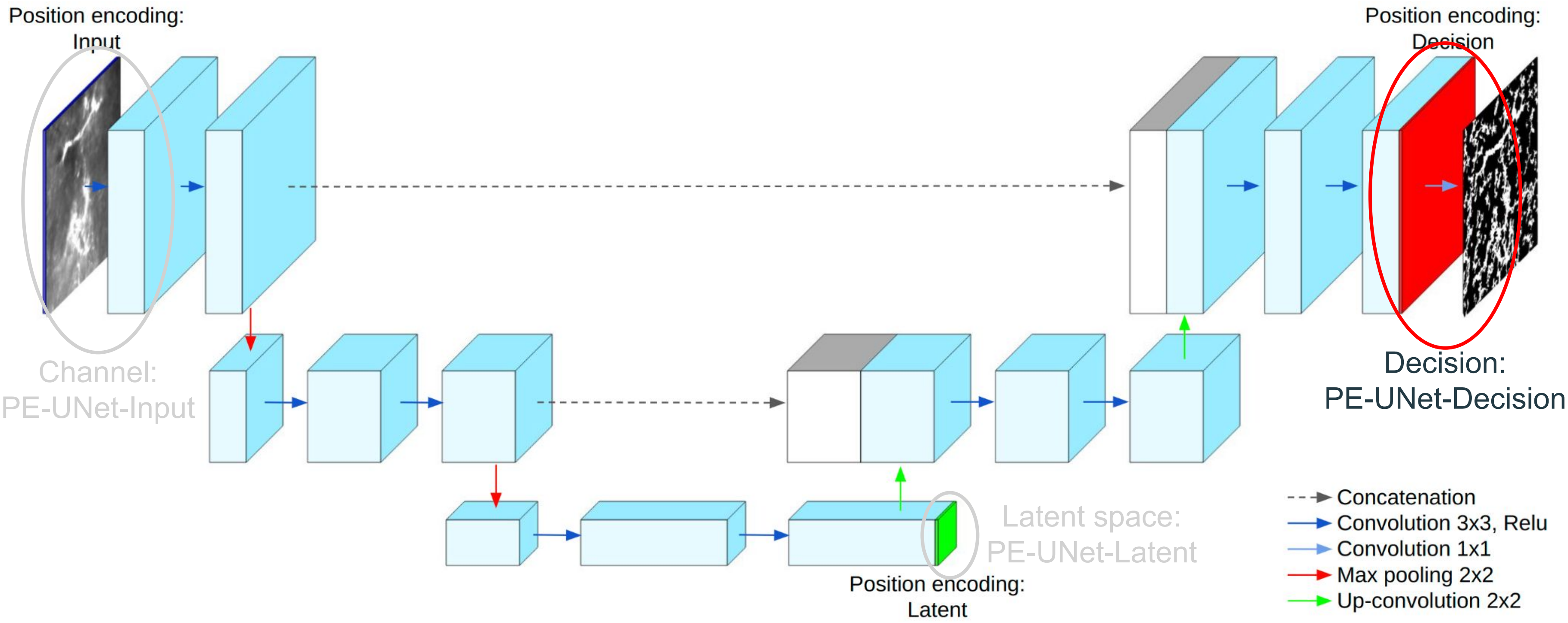
PE-UNets, Berthelot+, in press

PE-UNet-Latent



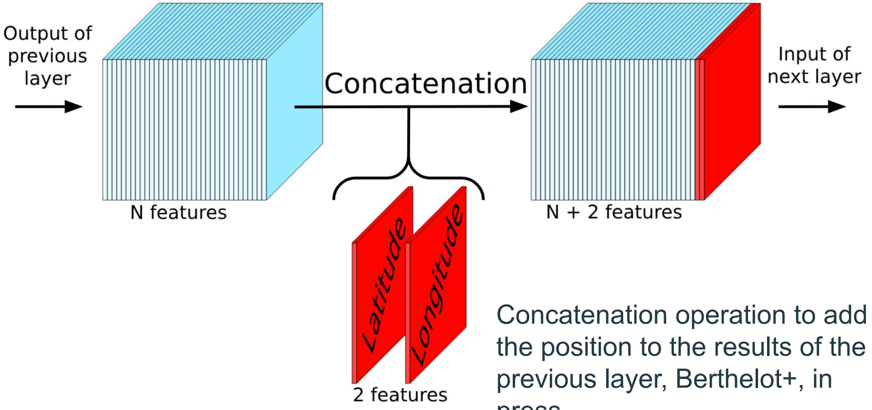
PE-UNets, Berthelot+, in press

PE-UNet-Decision



PE-UNets, Berthelot+, in press

How to add the position information?



Concatenation operation to add the position to the results of the previous layer, Berthelot+, in press

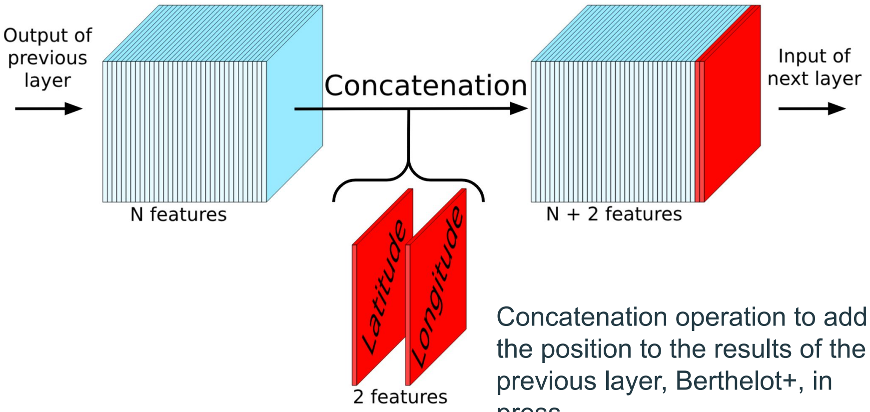
- Add the position information:
  - Fill one feature with latitude
  - Fill one feature with longitude
  - Concatenate them to the result of the previous layer

- Encode the position:
  - Data consistency
  - Handle outlier

Position encoding of the Galactic longitude

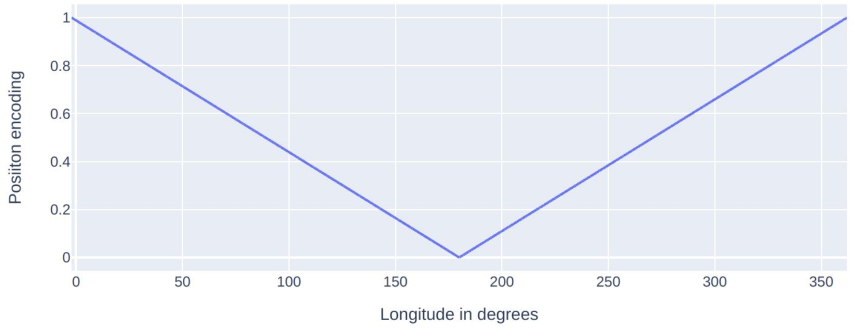
How to add the position information?

- Add the position information:
  - Fill one feature with latitude
  - Fill one feature with longitude
  - Concatenate them to the result of the previous layer



Concatenation operation to add the position to the results of the previous layer, Berthelot+, in press

- Encode the position:
  - Data consistency
  - Handle outlier



Position encoding of the Galactic longitude

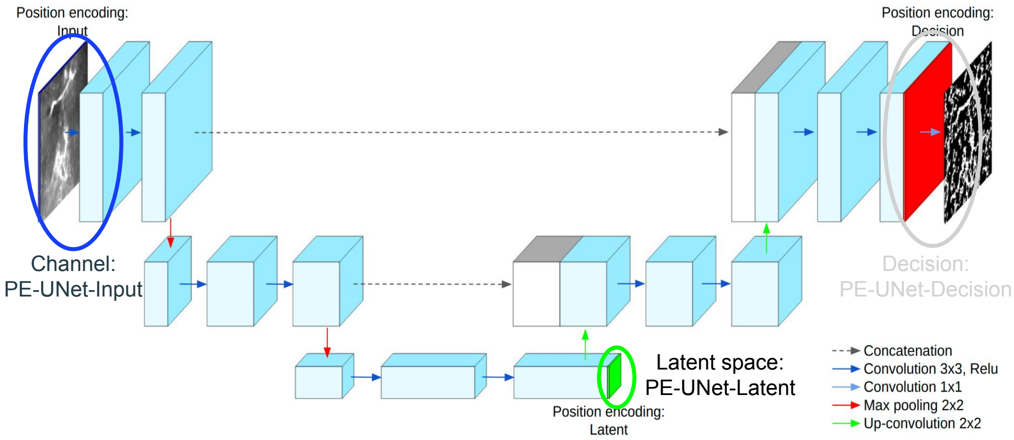
Putting some interpretability on PE-UNets

- Input/Latent:

Aim to extract long range relation between segmentation and position.

- Decision:

The PE-UNet-D perform segmentation regardless of the position then take it into consideration, acting like an adaptive threshold.



PE-UNets, Berthelot+, in press

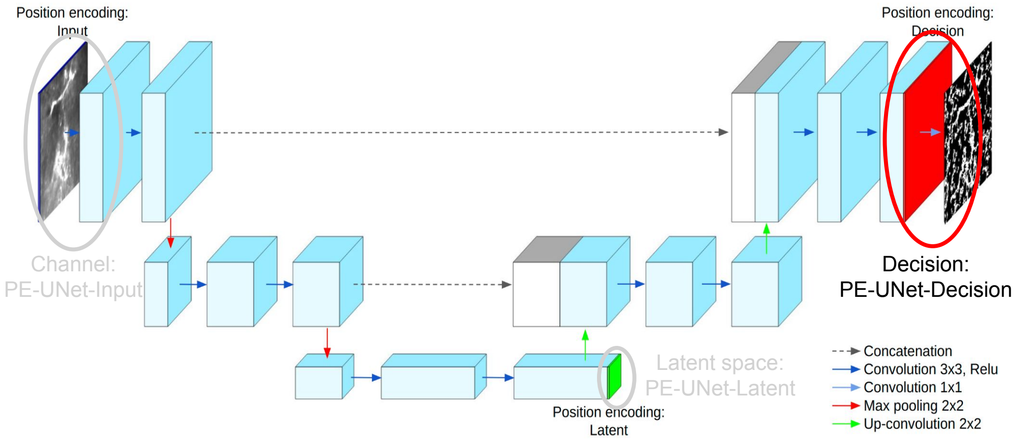
Putting some interpretability on PE-UNets

- Input/Latent:

Aim to extract long range relation between segmentation and position.

- Decision:

The PE-UNet-D perform segmentation regardless of the position then take it into consideration, acting like an *adaptive threshold*.



PE-UNets, Berthelot+, in press

Improving PE-UNet performance with data augmentaiton

How to improve PE-UNet performance

- Data augmentation improves model performance:
    - Flip
    - Rotation
    - Blur
    - Noise
    - Crop

⚠ Data augmentation should not change the label/reduce information

- Because of the filament-position relation, data augmentation could change the label

How to improve PE-UNet performance

Region 1

Data augmentation: flips and rotations

Region 1

Region 2

Data augmentation: flips and rotations

Region 2

Improving PE-UNet with Data augmentation

- Cross validation with 5 folds.
- Fixed learning rate, batch size, optimiser, etc.
- One cross-validation with no data augmentation strategy.
- One cross-validation with data augmentation (flips and rotations).
- Data augmentation virtually increases the dataset size by 16.
- Metrics used: DSC (IoU), mAP and AUC ROC

Can we improve PE-UNet performance with data augmentation?

→ PE-UNet-L performs better with DA

⚠ Maybe two effects play against each other:

- DA improving performance
- Physics non-reality decreasing performance

- We perform the same experiment with a UNet:
  - Performance gain is the same

→ DA is needed to increase performance

Model	DA	DSC	mAP	AUC ROC
UNet	no	0.9270	0.9551	0.9782
PE-UNet-L	no	0.9338	0.9627	0.9819
UNet	yes	0.9680	0.9949	0.9960
PE-UNet-L	yes	0.9746	0.9970	0.9976

Average metric value across the 5 folds, Berthelot+, in press

Can we improve PE-UNet performance with data augmentation?

→ PE-UNet-L performs better with DA

⚠ Maybe two effects play against each other:

- DA improving performance
- Physics non-reality decreasing performance

- We perform the same experiment with a UNet:
  - Performance gain is the same

Model	DA	DSC	mAP	AUC ROC
UNet	no	0.9270	0.9551	0.9782
PE-UNet-L	no	0.9338	0.9627	0.9819
UNet	yes	0.9680	0.9949	0.9960
PE-UNet-L	yes	0.9746	0.9970	0.9976

Average metric value across the 5 folds, Berthelot+, in press

→ DA is needed to increase performance

Can we improve PE-UNet performance with data augmentation?

→ PE-UNet-L performs better with DA

⚠ Maybe two effects play against each other:
- DA improving performance
- Physics non-reality decreasing performance

- We perform the same experiment with a UNet:
  - Performance gain is the same

Model	DA	DSC	mAP	AUC ROC
UNet	no	0.9270	0.9551	0.9782
PE-UNet-L	no	0.9338	0.9627	0.9819
UNet	yes	0.9680	0.9949	0.9960
PE-UNet-L	yes	0.9746	0.9970	0.9976

Average metric value across the 5 folds, Berthelot+, in press

→ DA is needed to increase performance

Can we improve PE-UNet performance with data augmentation?

→ PE-UNet-L performs better with DA

⚠ Maybe two effects play against each other:
  - DA improving performance
  - Physics non-reality decreasing performance
- We perform the same experiment with a UNet:
  - Performance gain is the same

Model	DA	DSC	mAP	AUC ROC
UNet	no	0.9270	0.9551	0.9782
PE-UNet-L	no	0.9338	0.9627	0.9819
UNet	yes	0.9680	0.9949	0.9960
PE-UNet-L	yes	0.9746	0.9970	0.9976

Average metric value across the 5 folds, Berthelot+, in press

→ **DA** is **needed** to increase performance

Cross validation and model comparison

Experiment settings

- 5 folds
- Hyper-parameter: Learning rate
- Data augmentation: Flips and rotations (x16)
- Optimizer: Adam
- Models used:
  - UNet
  - UNet++
  - SwinUNet
  - PE-UNet(s)
- Metrics used: DSC (IuO), mAP, AUC ROC

PE-UNet improves state of the art

PE-UNet-Latent gets best average score for the 3 metrics

Model	DSC	mAP	AUC ROC
UNet	0.9680	0.9949	0.9960
UNet++	0.9690	0.9953	0.9960
SwinUNet	0.9637	0.9939	0.9950
PE-UNet-I	0.9649	0.9950	0.9960
PE-UNet-D	0.9685	0.9952	0.9962
PE-UNet-L	**0.9746**	**0.9970**	**0.9976**

Average metric value across the 5 folds, Berthelot+, in press

PE-UNet's improvements are significant

- We perform statistical test between models
- A p-value lower than 0.05 (values in bold) indicates a confidence of 95% than the difference is significant
- The PE-UNet-L is significantly better than every other models except for the PE-UNet-I

Method	PE-UNet-L
UNet	**0.0175**
UNet++	**0.0061**
SwinUNet	**0.0031**
PE-UNet-D	**0.0006**
PE-UNet-I	0.1676

P-value of statistical test between models, Berthelot+, in press

How do metrics relate to maps?

Model	DSC
PE-UNet-L	**0.9746**

Average DSC for PE-UNet-L,
Berthelot+, in press

- About 97% labeled pixels are correctly classified by our model (PE-UNet-L)

- For the unknown pixels, an empirical study has been done showing that our model performs well

Background pixels are in green, filament labels in red and filament segmentation by our PE-UNet-L in blue
Berthelot+, in press

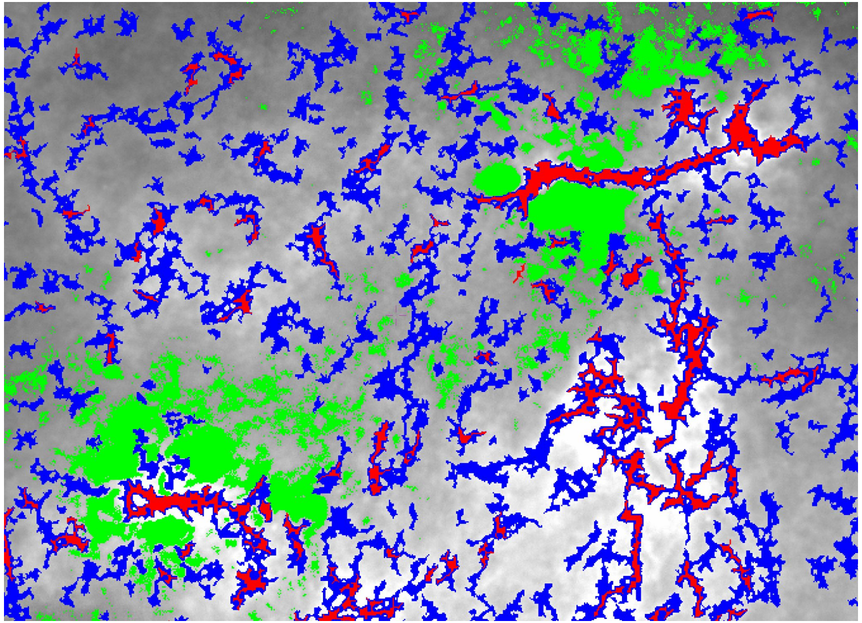
How do metrics relate to maps?

Model	DSC
PE-UNet-L	**0.9746**

Average DSC for PE-UNet-L,
Berthelot+, in press

- About 97% labeled pixels are correctly classified by our model (PE-UNet-L)

- For the unknown pixels, an empirical study has been done showing that our model performs well



Background pixels are in green, filament labels in red and filament segmentation by our PE-UNet-L in blue
Berthelot+, in press

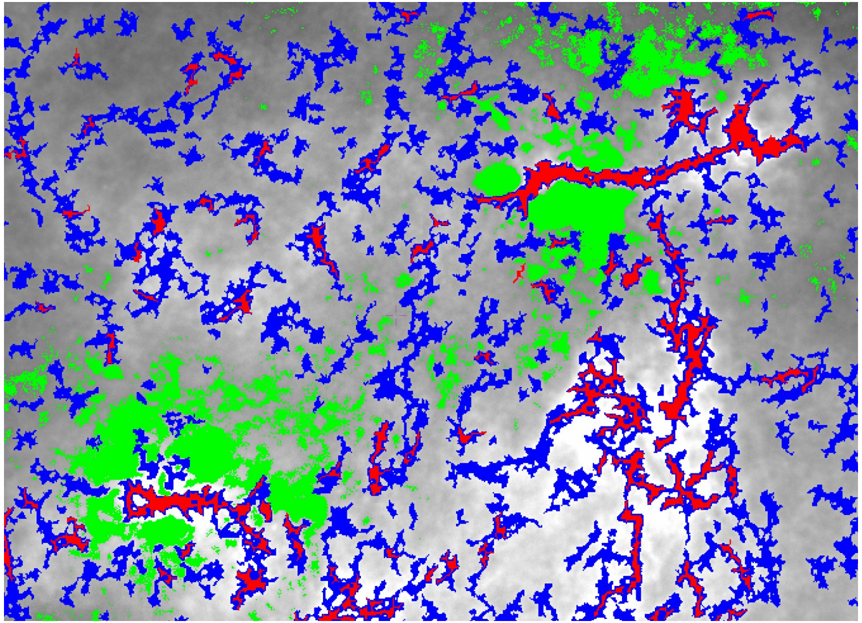
How do metrics relate to maps?

Model	DSC
PE-UNet-L	**0.9746**

Average DSC for PE-UNet-L,
Berthelot+, in press

- About 97% labeled pixels are correctly classified by our model (PE-UNet-L)

- For the unknown pixels, an empirical study has been done showing that our model performs well

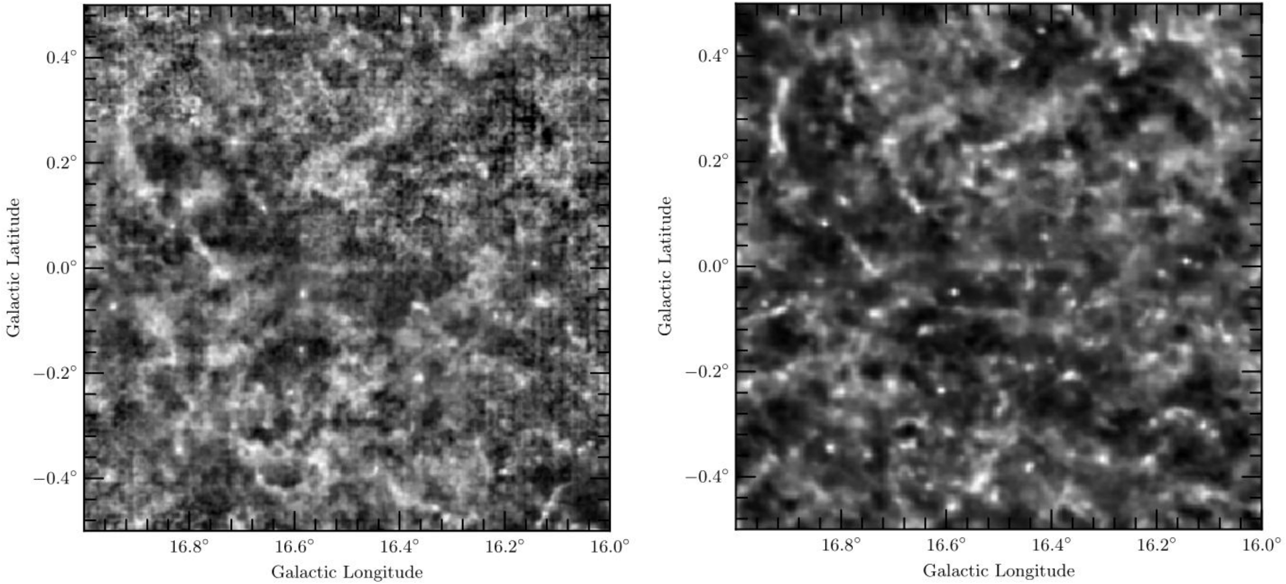


Background pixels are in green, filament labels in red and filament segmentation by our PE-UNet-L in blue
Berthelot+, in press

Applying our model to COHRS data ^{12}CO (3-2) survey

Segmentation of COHRS (CO survey of the Gp)

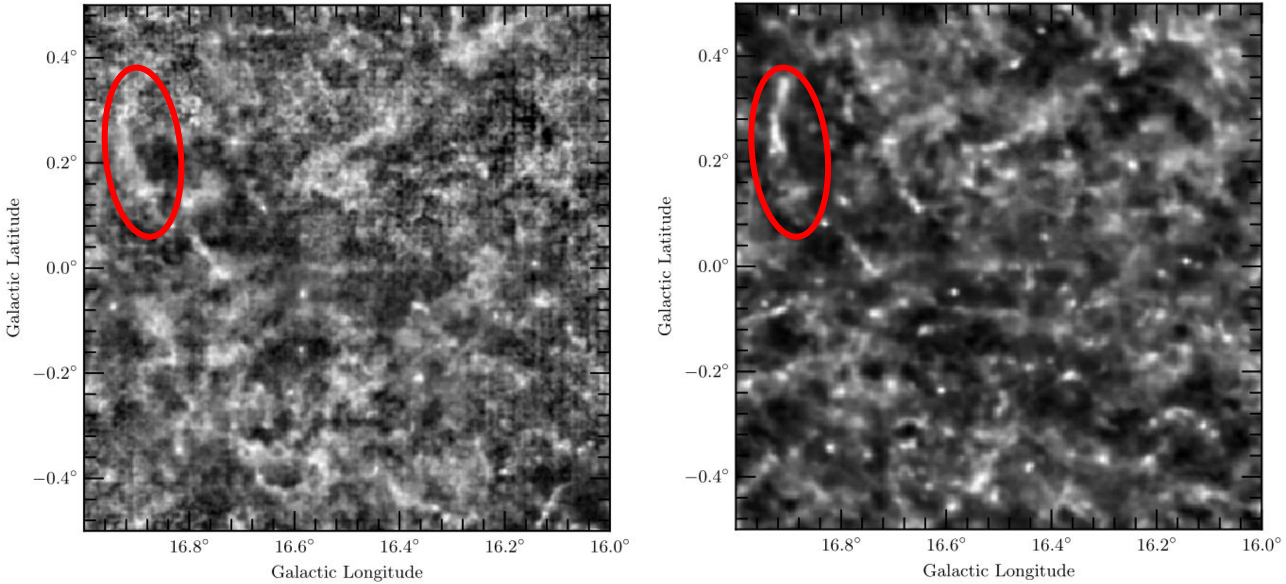
- COHRS: ^{12}CO (3-2) survey
- Hi-GAL and COHRS don't trace the same physical conditions



COHRS normed image on the left, Hi-GAL normed image on the right, Berthelot+ in press

Segmentation of COHRS (CO survey of the Gp)

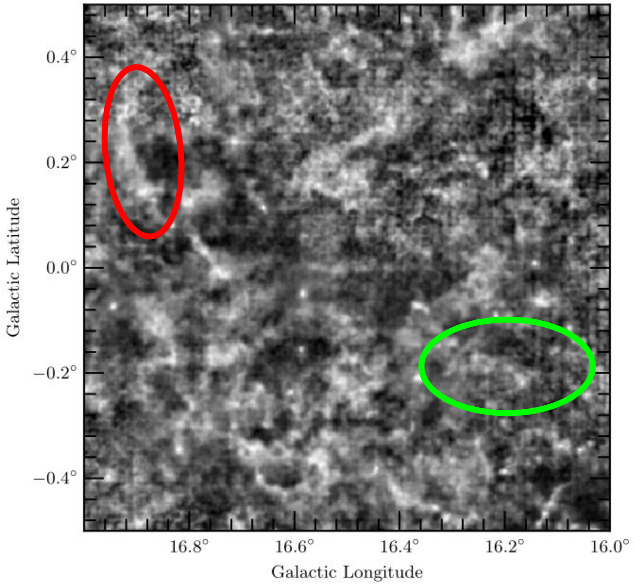
- COHRS: ^{12}CO (3-2) survey
- Hi-GAL and COHRS don't trace the same physical conditions

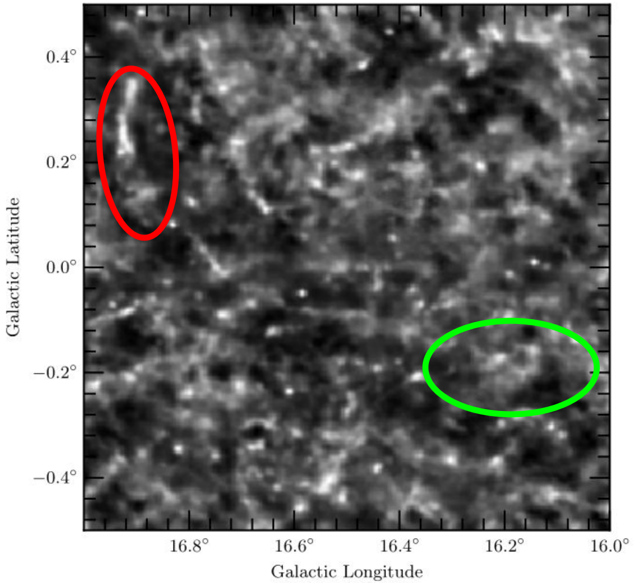


COHRS normed image on the left, Hi-GAL normed image on the right, Berthelot+ in press

Segmentation of COHRS (CO survey of the Gp)

- COHRS: ^{12}CO (3-2) survey
- Hi-GAL and COHRS don't trace the same physical conditions

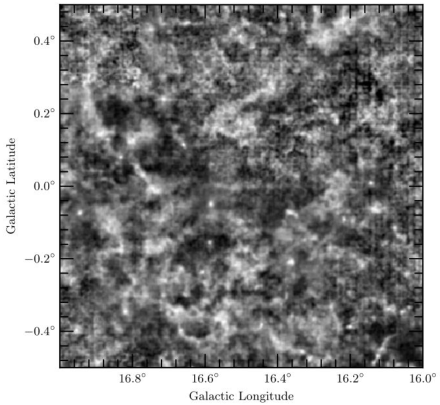


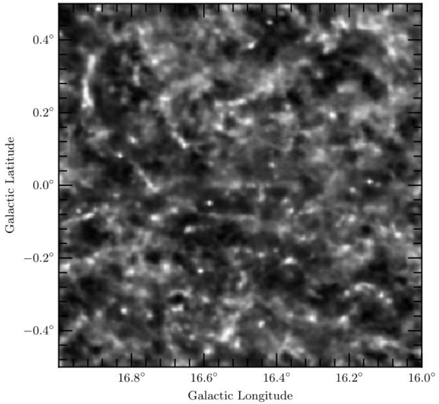


COHRS normed image on the left, Hi-GAL normed image on the right, Berthelot+ in press

Segmentation of COHRS (CO survey of the Gp)

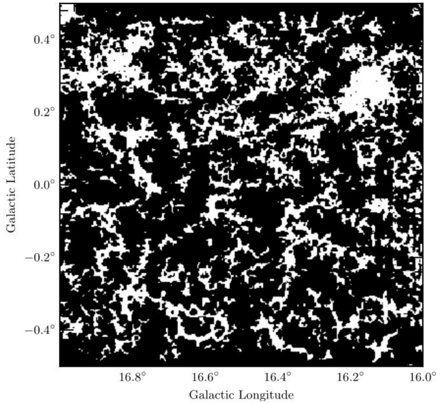
$^{12}CO(3\text{-}2)$ COHRS normed image

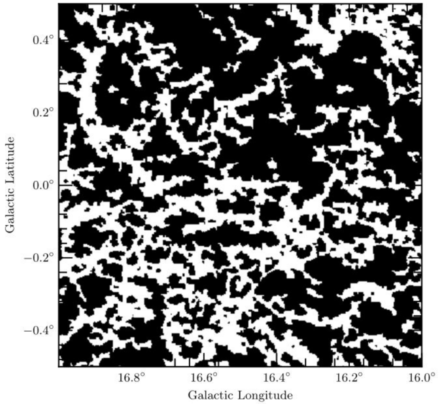




Hi-GAL normed image

$^{12}CO(3\text{-}2)$ COHRS Segmentation

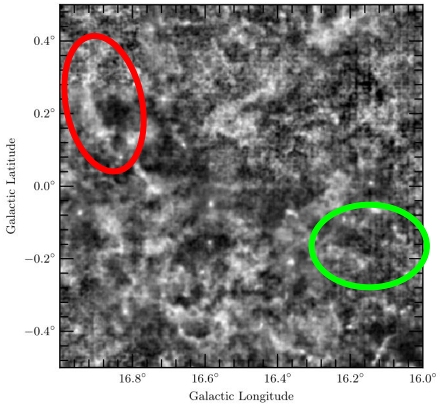




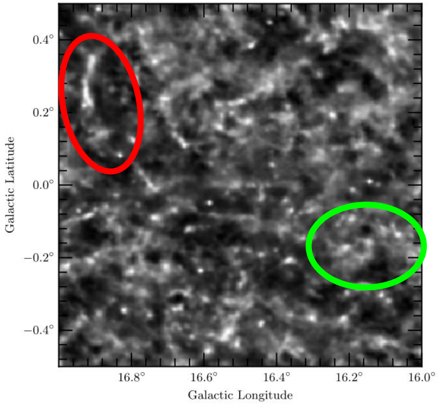
Hi-GAL segmentation

Segmentation of COHRS (CO survey of the Gp)

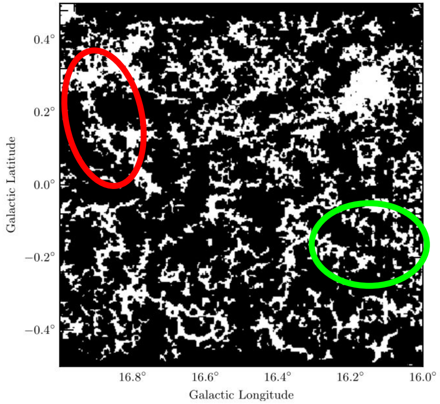
$^{12}CO(3\text{-}2)$ COHRS normed image



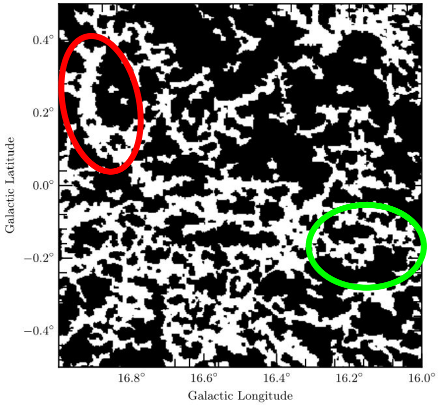
Hi-GAL normed image



$^{12}CO(3\text{-}2)$ COHRS Segmentation

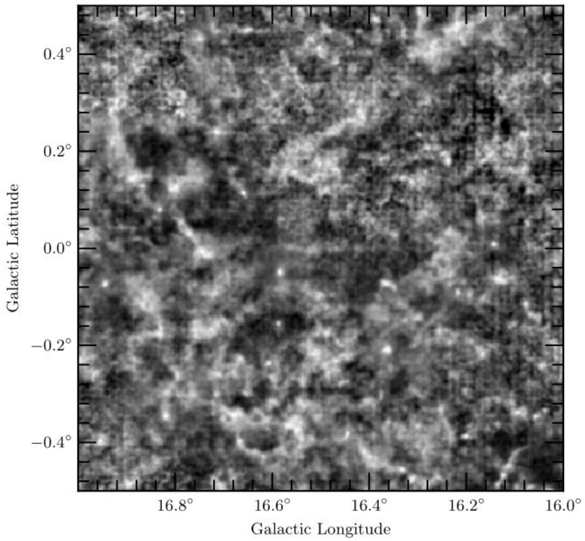


Hi-GAL segmentation



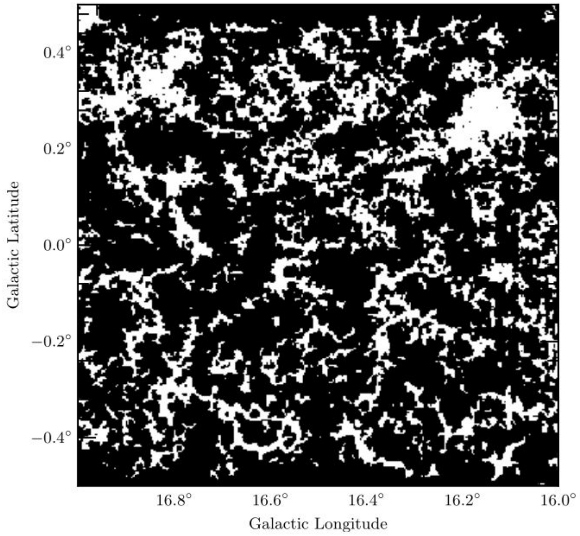
Segmentation of COHRS (CO survey of the Gp)

- Our model is able to detect filaments in COHRS
- No reference exists on COHRS for filament extraction
- COHRS presents very noisy images



COHRS normed image

PE-UNet-L trained on Hi-GAL



COHRS segmentation

Conclusions

Conclusions

- Cross-validation scheme on Hi-GAL dataset
  - Patch learning strategy
  - Normalisation
  - Semi-supervised learning

- PE-UNet improves significatively the state of the art

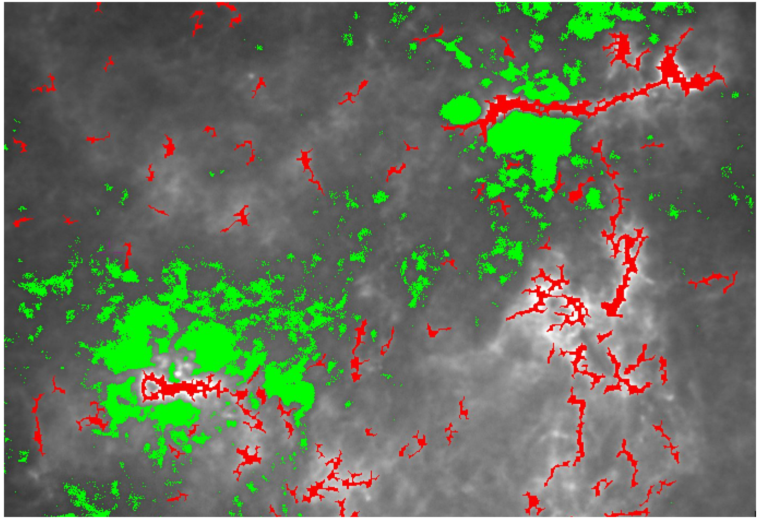
- Application of our models to other datasets
  - COHRS

Perspectives

⚠ From a machine learning standpoint, performance validation is the biggest challenge

→ Work towards ways to validate filament candidates

- Using 3D data from observations
- Using numerical simulations



H_2 column density: Hi-GAL dataset, Pixel labels, background pixels are in green and filament pixels are in red. Rest is unknown.

Thanks for your attention