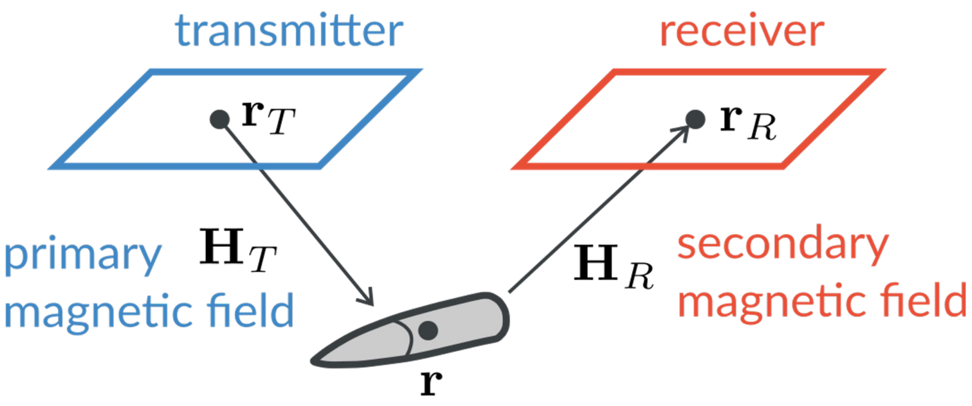
Using convolutional neural networks to classify UXO with multi-component electromagnetic induction data

Jorge Lopez-Alvis[1], Lindsey J. Heagy[1], Douglas W. Oldenburg[1], Stephen Billings[2], Lin-Ping Song[2]

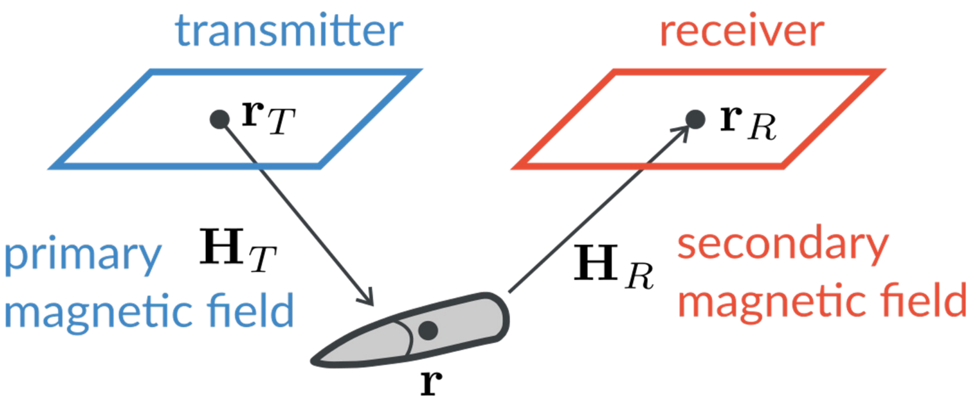
[1]University of British Columbia, [2]Black Tusk Geophysics, Inc.

This work is supported by DoD SERDP project MR22-3487

Time-domain EM response of a UXO



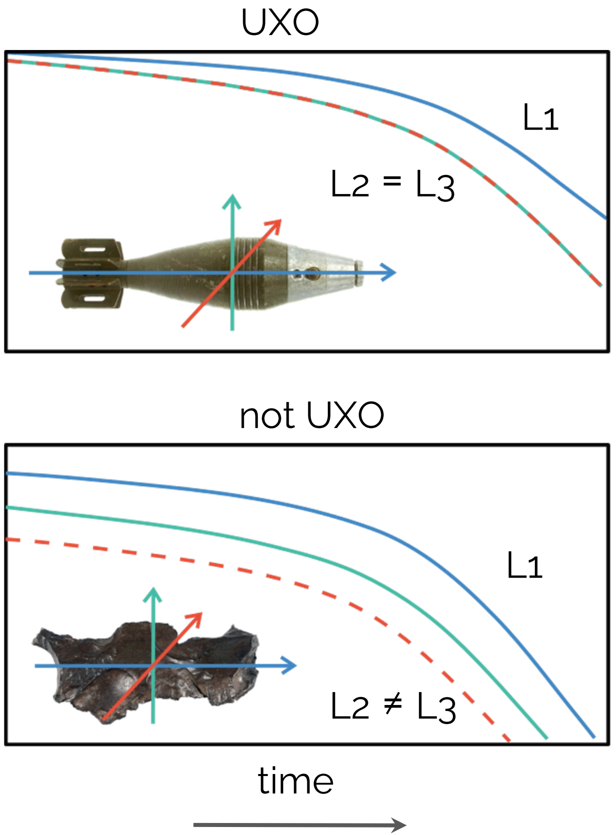
Time-domain EM response of a UXO



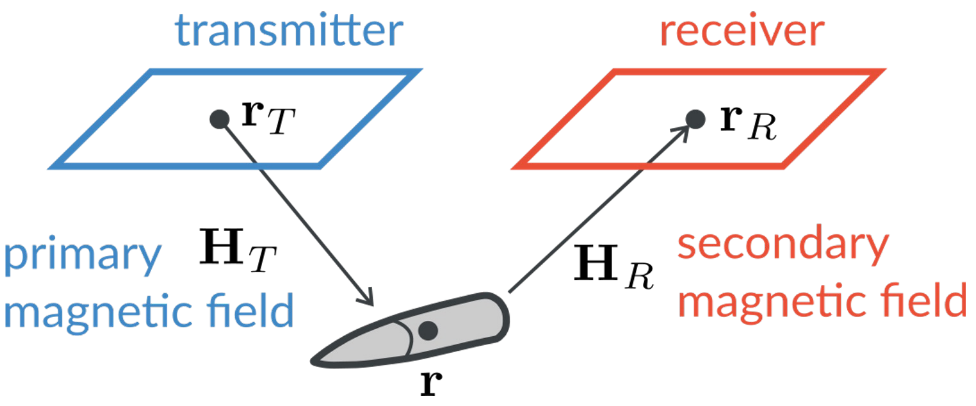
$$d(\mathbf{r}_R, t) = \mathbf{H}_R(\mathbf{r}, \mathbf{r}_R) \cdot \mathbf{P}(t) \cdot \mathbf{H}_T(\mathbf{r}, \mathbf{r}_T)$$

$$\mathbf{P}(t) = \mathbf{A}(\phi, \theta, \psi) \cdot \mathbf{L}(t) \cdot \mathbf{A}^\top(\phi, \theta, \psi)$$

$$\mathbf{L}(t) = \begin{pmatrix} L_1 & & \\ & L_2 & \\ & & L_3 \end{pmatrix}$$



Time-domain EM response of a UXO

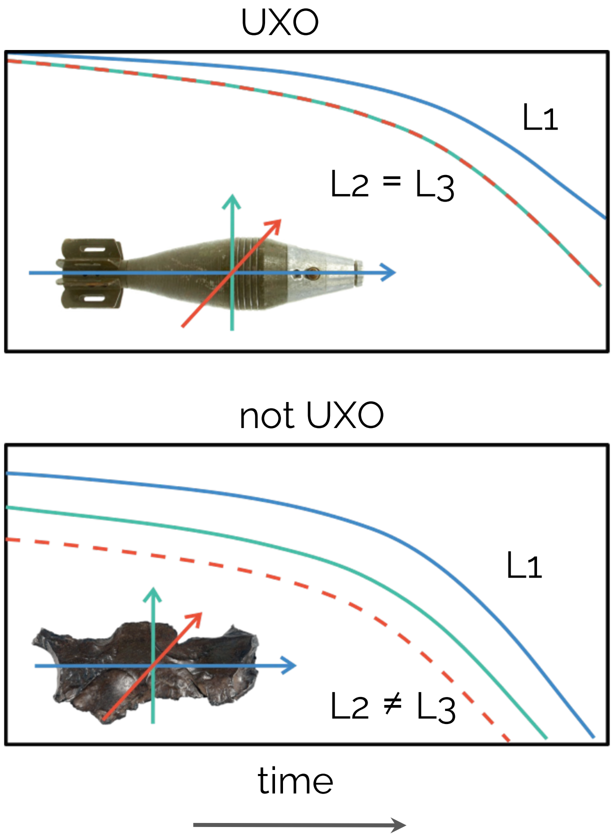


$$d(\mathbf{r}_R, t) = \mathbf{H}_R(\mathbf{r}, \mathbf{r}_R) \cdot \mathbf{P}(t) \cdot \mathbf{H}_T(\mathbf{r}, \mathbf{r}_T)$$

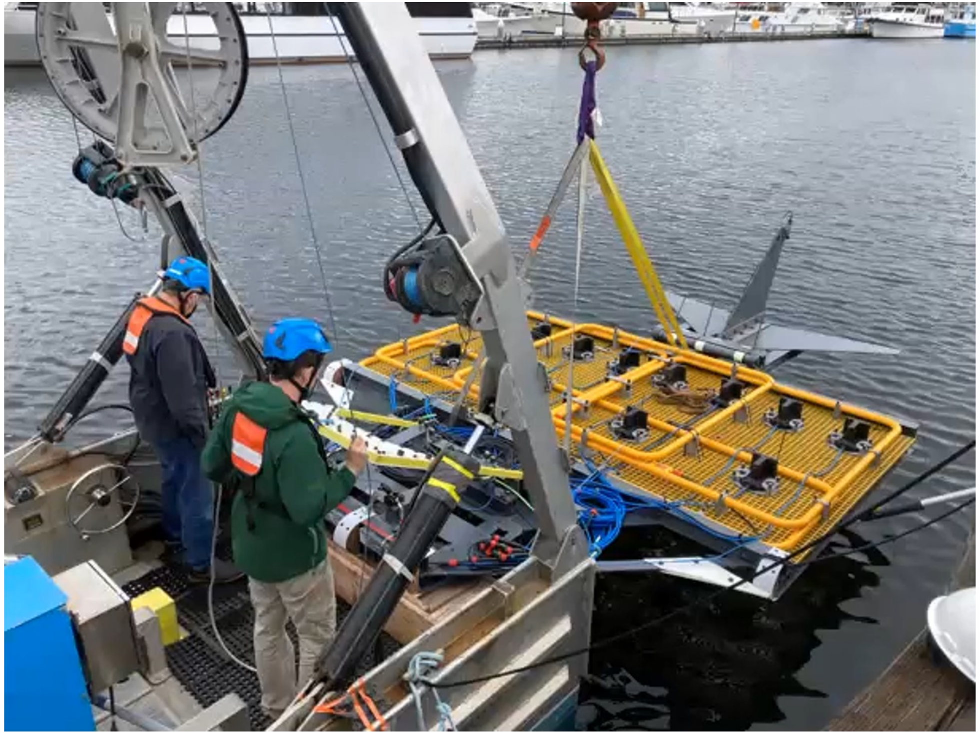
$$\mathbf{P}(t) = \mathbf{A}(\phi, \theta, \psi) \cdot \mathbf{L}(t) \cdot \mathbf{A}^\top(\phi, \theta, \psi)$$

$$\mathbf{L}(t) = \begin{pmatrix} L_1 & & \\ & L_2 & \\ & & L_3 \end{pmatrix}$$

traditional approach: use inversion to get these and then classify by comparing **L**(t) with ordnance library



Survey and system



UltraTEMA-4 system:

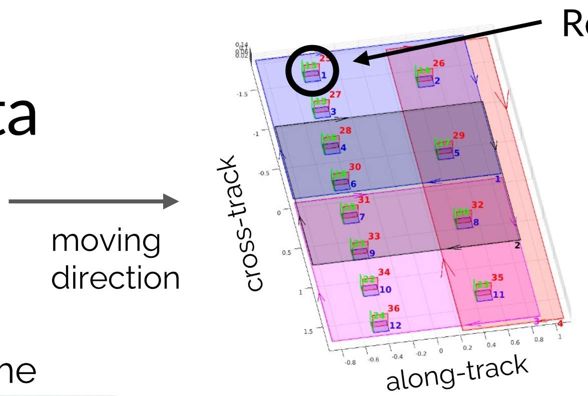
4 transmitters

12 receivers (3-component)

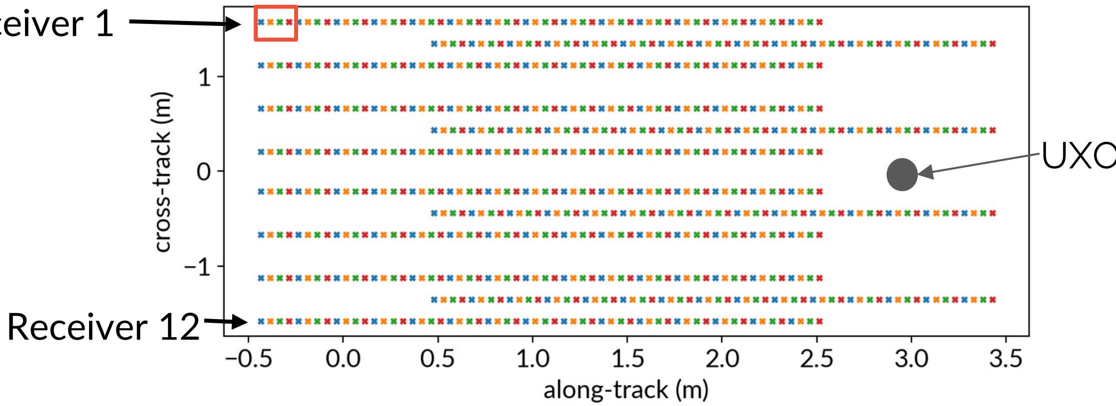
27 time channels

Height above seabed: ~1 m

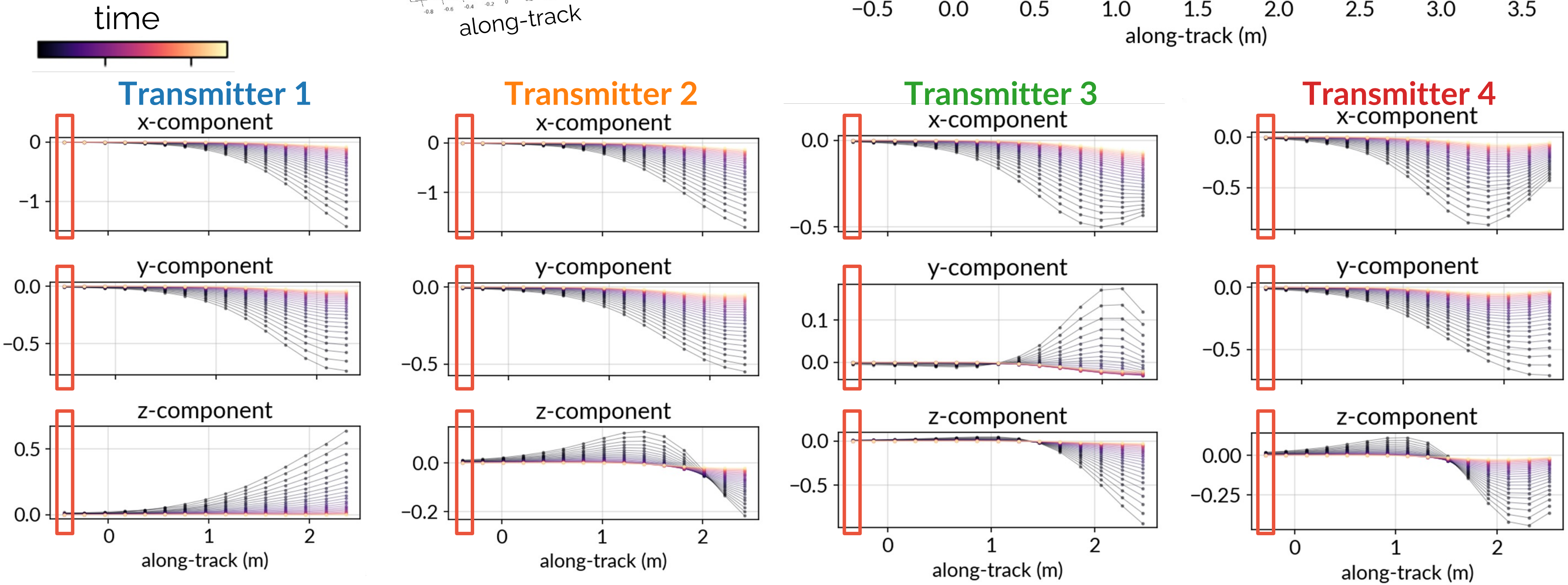
Data



moving
direction
cross-track
along-track

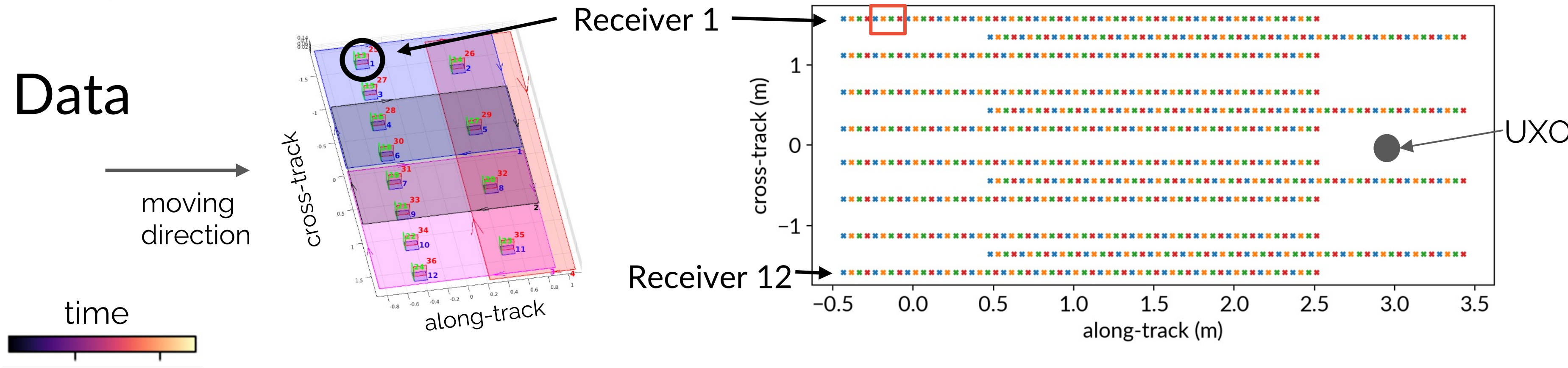


Receiver 1
Receiver 12
UXO
cross-track (m)
along-track (m)
1
0
−1
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

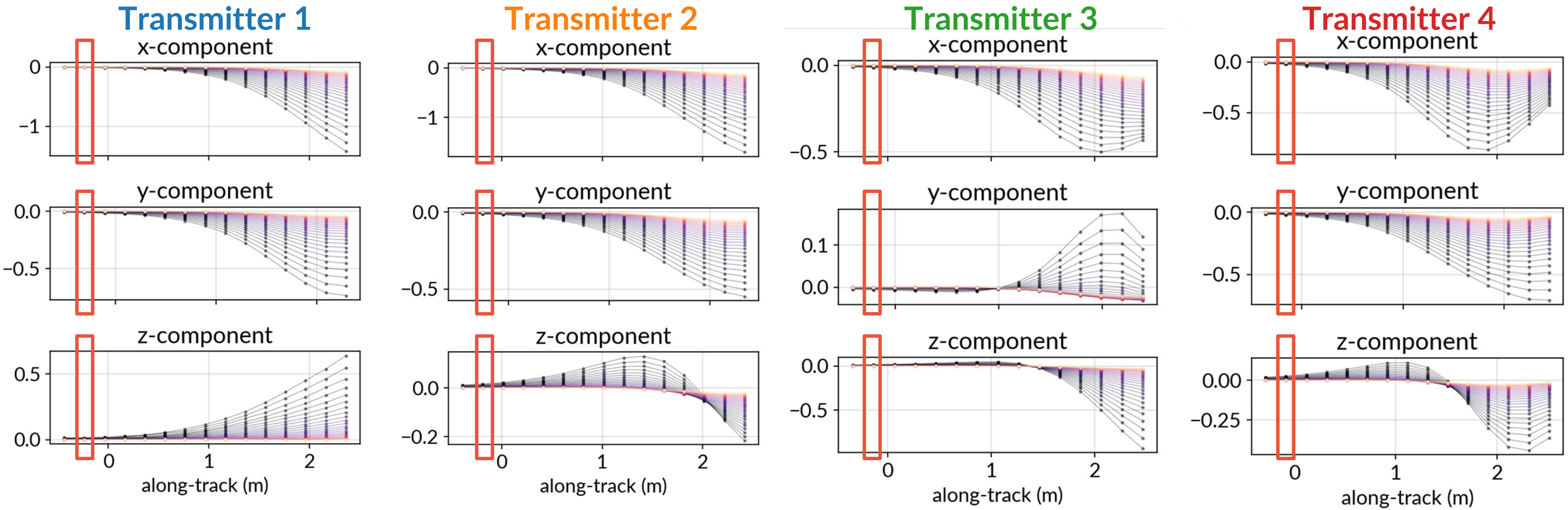


time
Transmitter 1
Transmitter 2
Transmitter 3
Transmitter 4
x-component
y-component
z-component
along-track (m)

Data

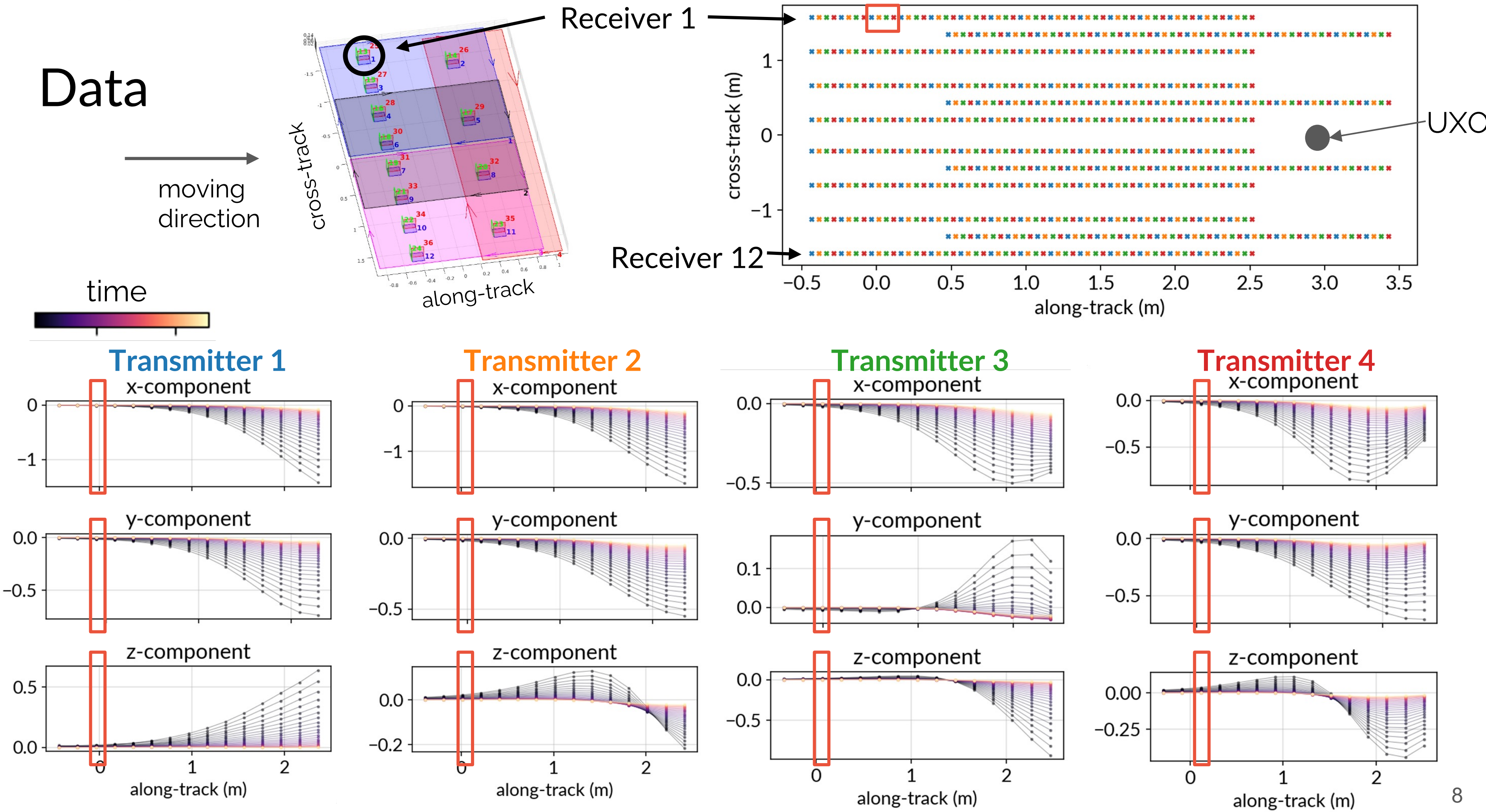


moving
direction
cross-track
along-track
time
Receiver 1
Receiver 12
UXO
cross-track (m)
along-track (m)

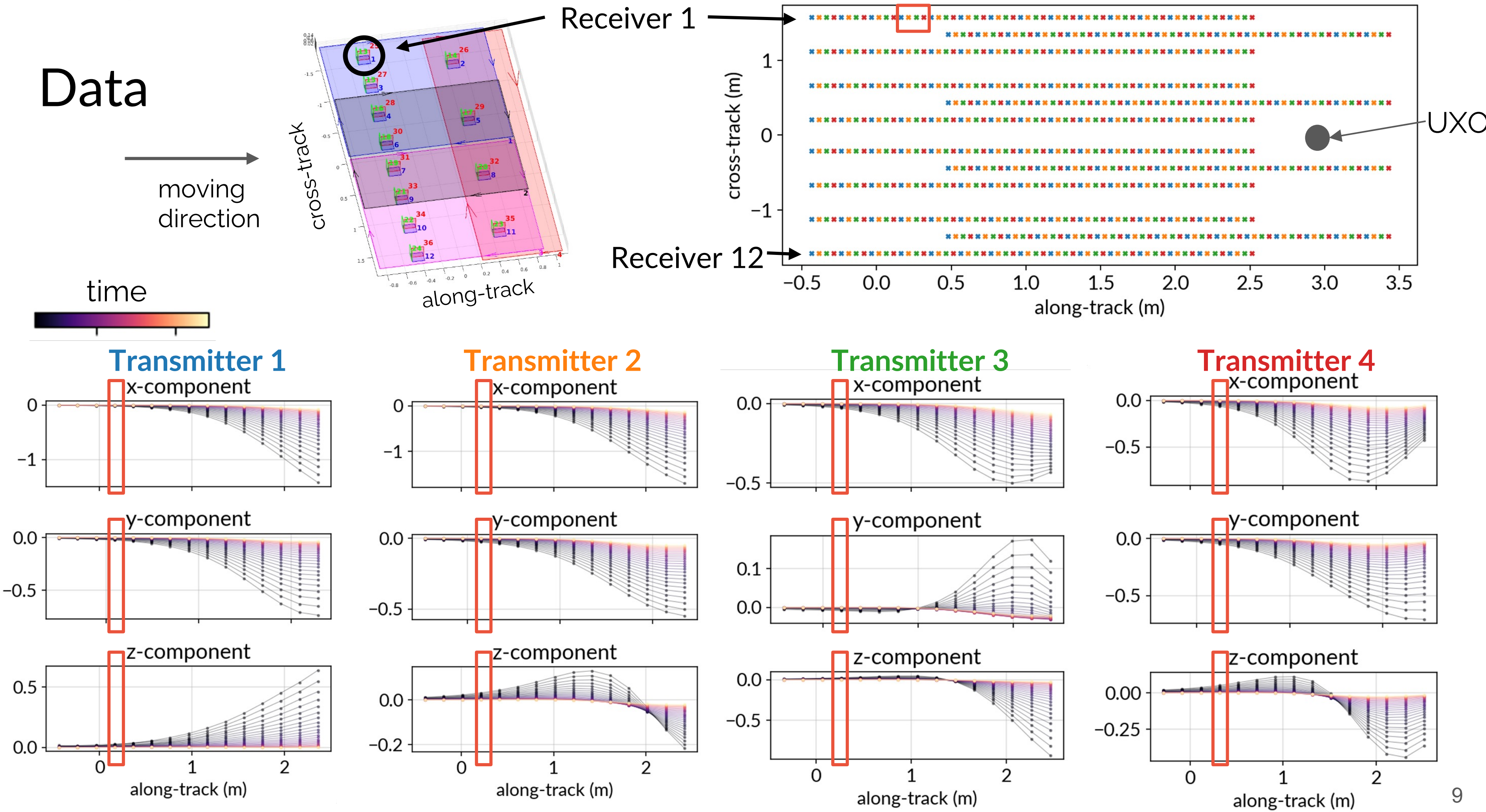


Transmitter 1
Transmitter 2
Transmitter 3
Transmitter 4
x-component
y-component
z-component
along-track (m)

Data


moving direction
cross-track
along-track
Receiver 1
Receiver 12
UXO
cross-track (m)
along-track (m)
time
Transmitter 1
Transmitter 2
Transmitter 3
Transmitter 4
x-component
y-component
z-component
along-track (m)

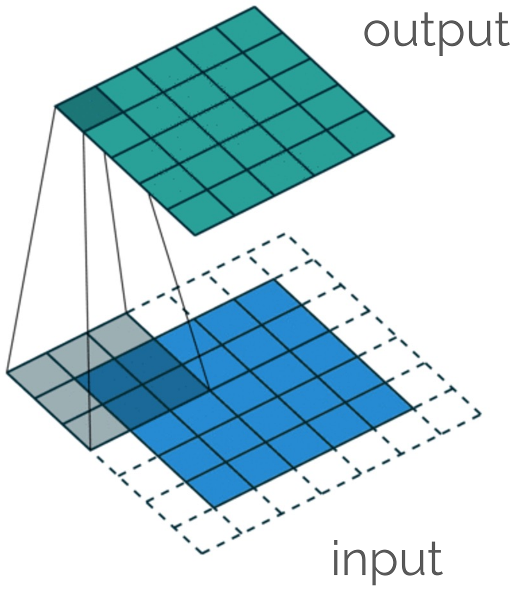
Data


moving
direction
cross-track
along-track
Receiver 1
Receiver 12
UXO
cross-track (m)
along-track (m)
time
Transmitter 1
Transmitter 2
Transmitter 3
Transmitter 4
x-component
y-component
z-component
along-track (m)

Can we classify directly from EM data?

Convolutional neural networks (CNNs)

- Convolutional filters look at spatial / temporal features in the data



Training EM data for UXO classification:

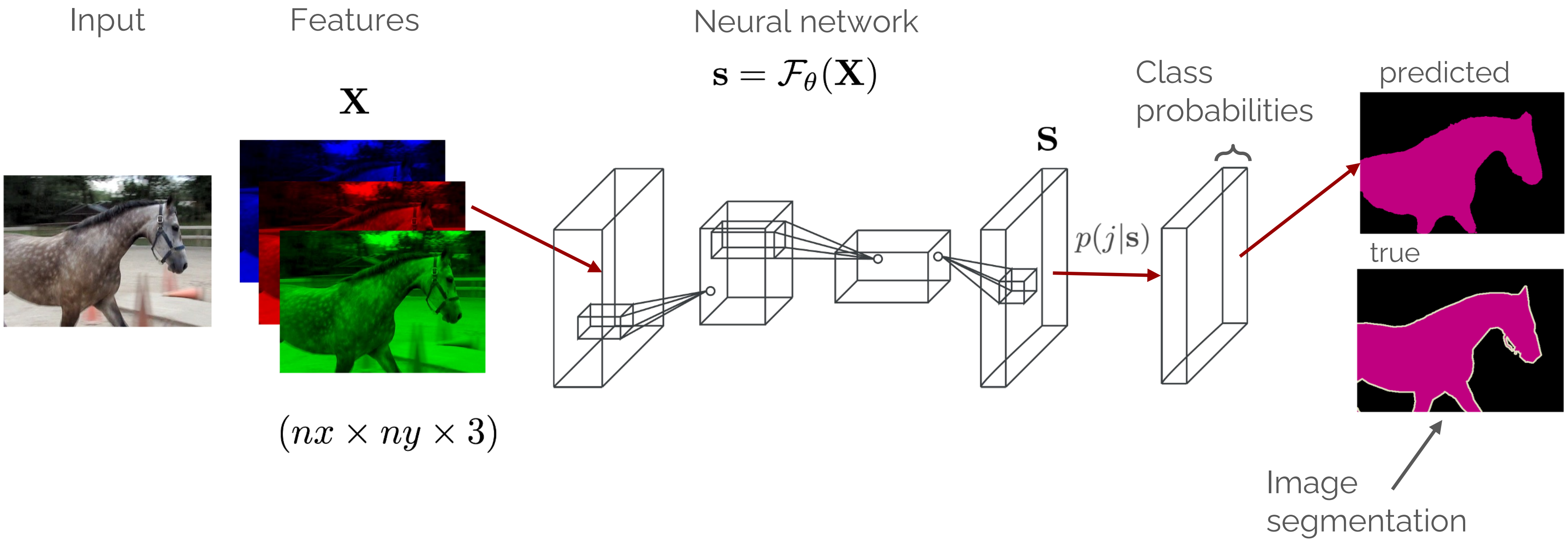
- Available library of ordnance objects with polarizations
- Fast geophysical simulations

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Convolutional Neural Networks (CNNs)

Supervised classification problem

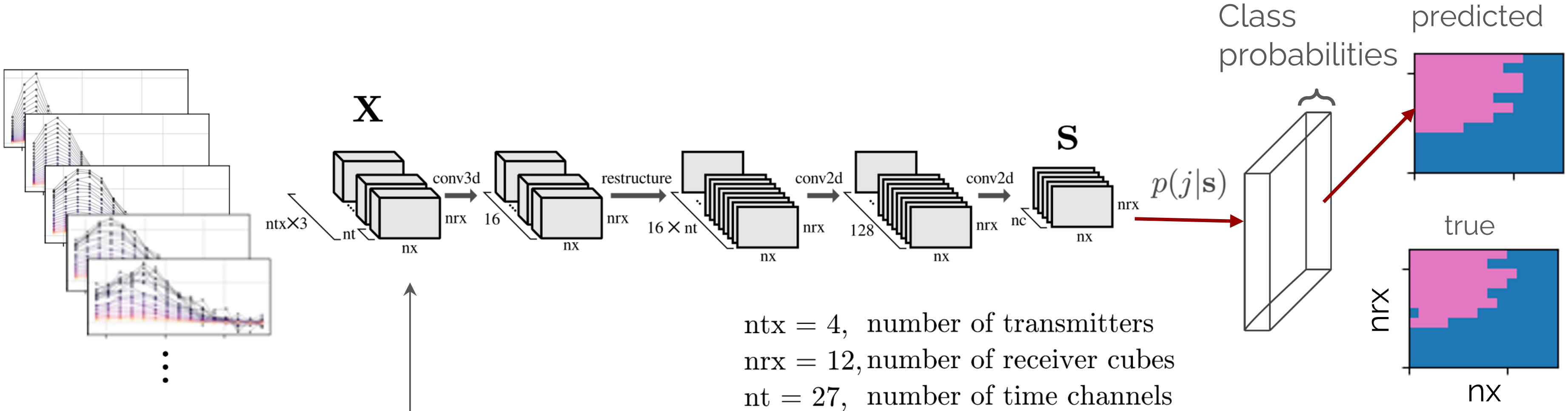
provided data with labels, construct a function (network) that outputs labels given input data



Convolutional Neural Networks (CNNs)

How do we translate these things to the UXO classification problem?

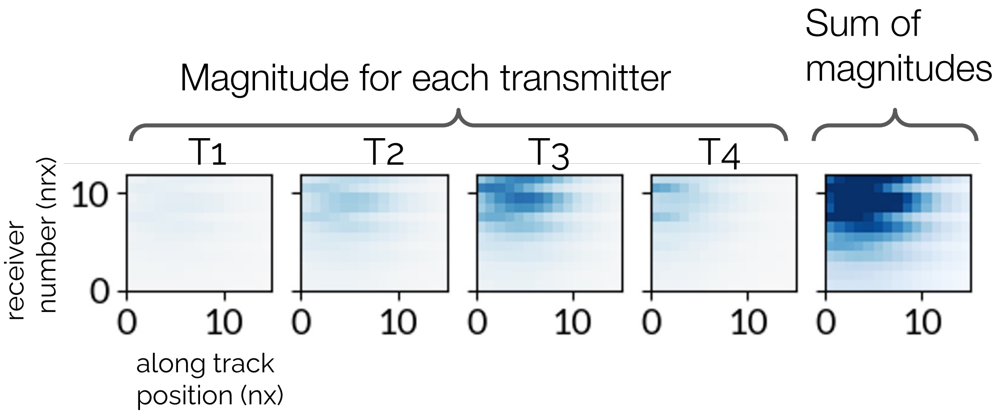
Input Features Neural network

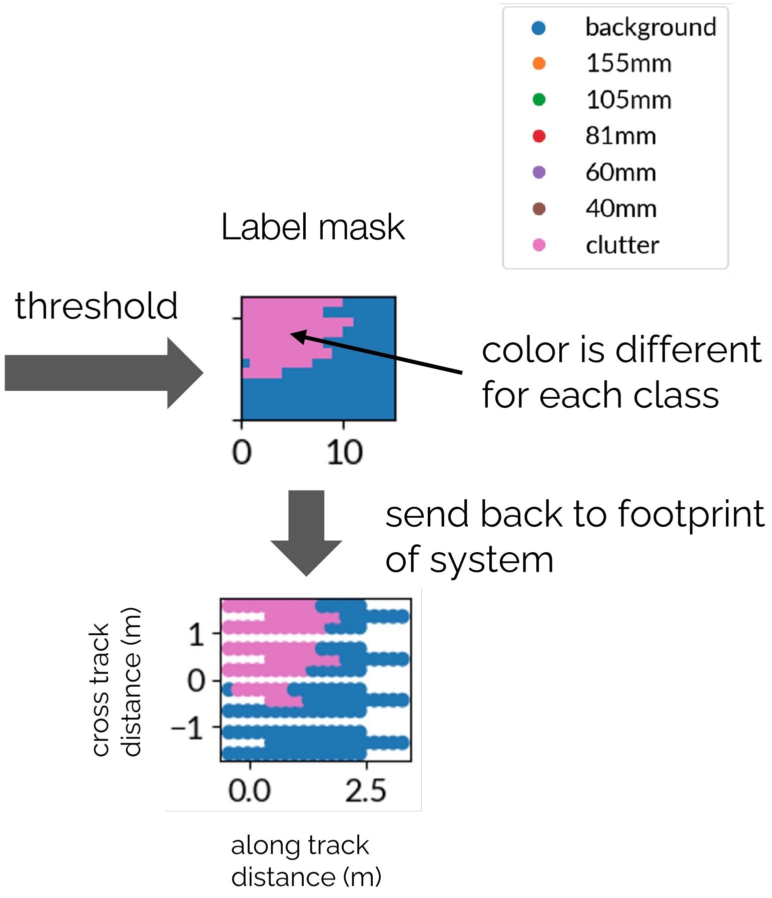


$$(nx \times nrx \times nt \times (ntx \times 3))$$

$ntx = 4,$ number of transmitters
$nrx = 12,$ number of receiver cubes
$nt = 27,$ number of time channels
$nx = 15,$ number of positions in spatial window (along track)
$nc = 8,$ number of classes

Defining label masks

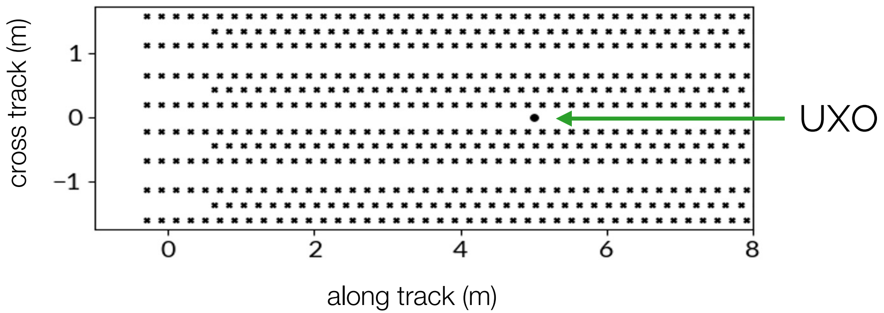




For time channel #5

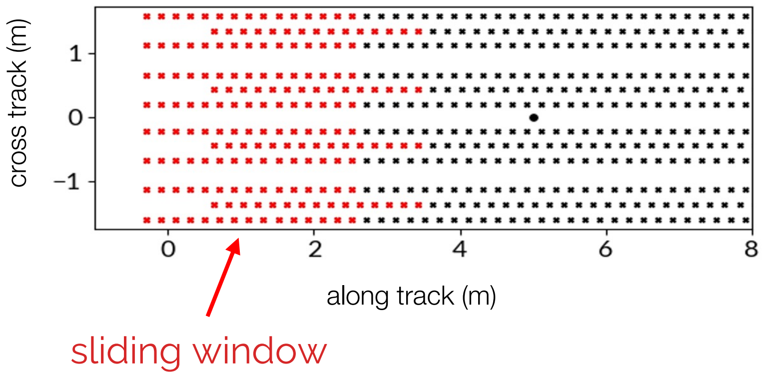
Application to a line of data

Input features are created by using a sliding window:



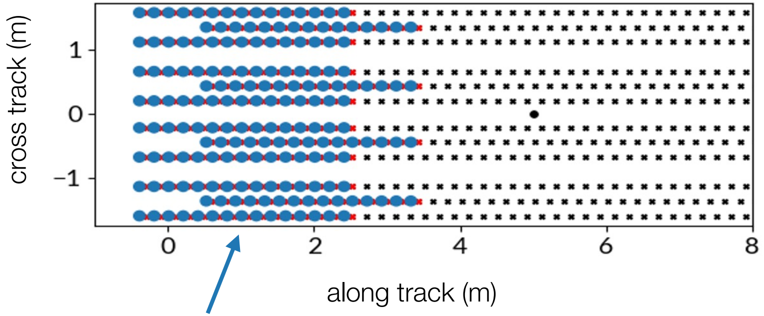
Application to a line of data

Input features are created by using a sliding window:



Application to a line of data

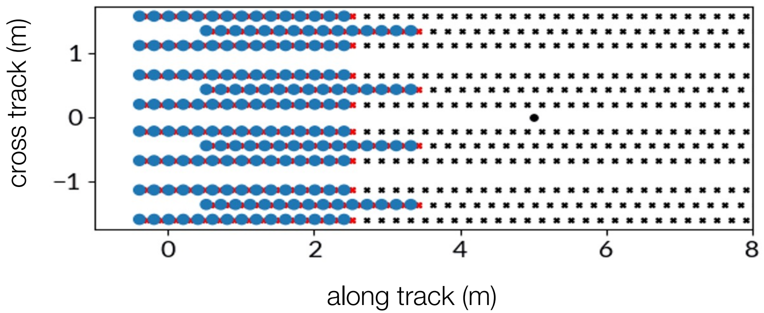
Input features are created by using a sliding window:



Neural network output (class)

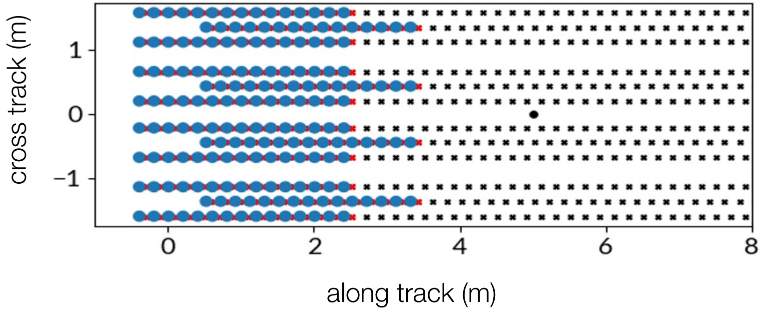
Application to a line of data

Input features are created by using a sliding window:

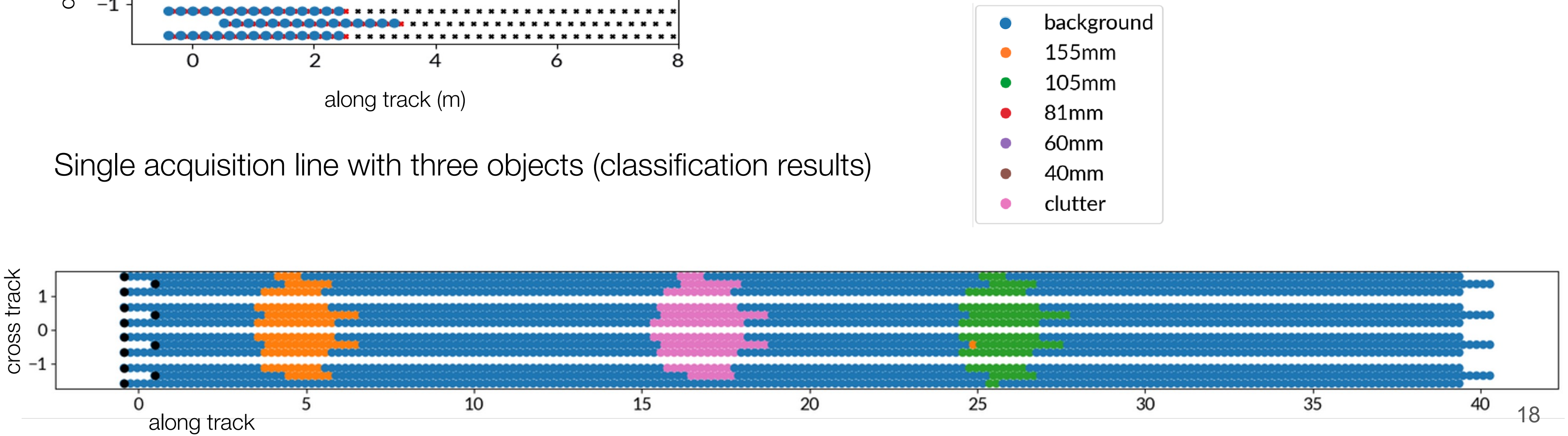


Application to a line of data

Input features are created by using a sliding window:



Single acquisition line with three objects (classification results)



Training dataset: dipole forward model

7 classes:

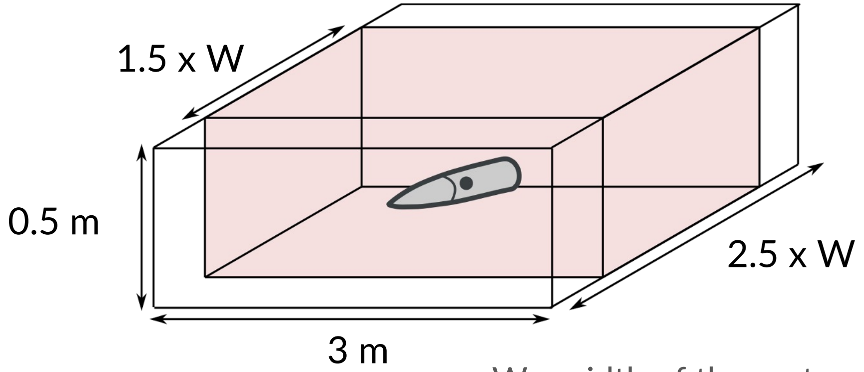
- background
- 155 mm
- 105 mm
- 81 mm
- 60 mm
- 40 mm
- clutter

# of realizations:

- Training (multi-class): 400,000
- Validation: 10,000

Randomly assign:

- Target class
- Location (x, y, z)
- Orientation (ϕ, θ, ψ)
- Noise level: approximate from background areas in the field data



W - width of the system

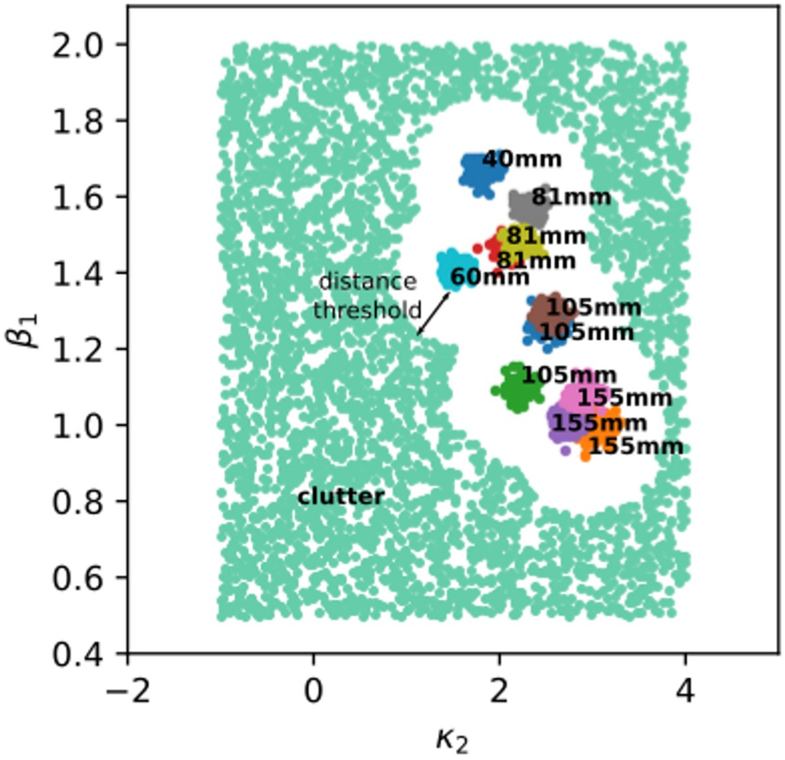
Clutter design

Physics-based parameterization of EM decay:

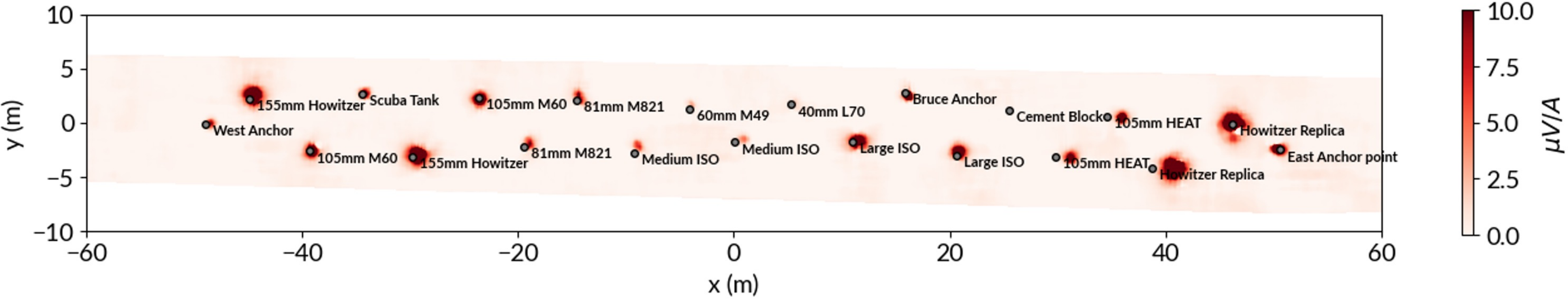
$$L(t) = kt^{-\beta} exp(- t/\gamma)$$

9 parameters in total:

1. Estimate values for UXOs in ordnance library
2. Define a distance threshold
3. Fill the remaining space with clutter objects

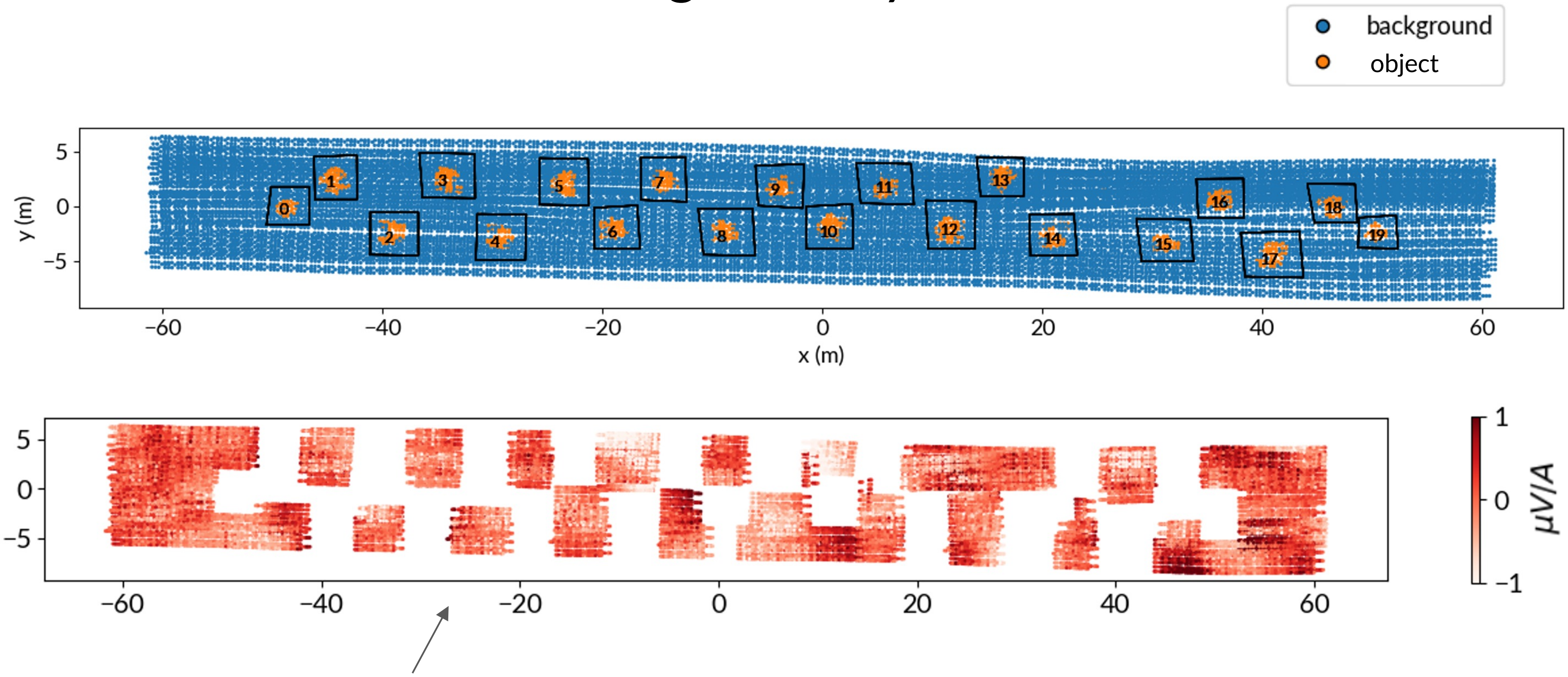


Field data - Sequim Bay test site (2022)



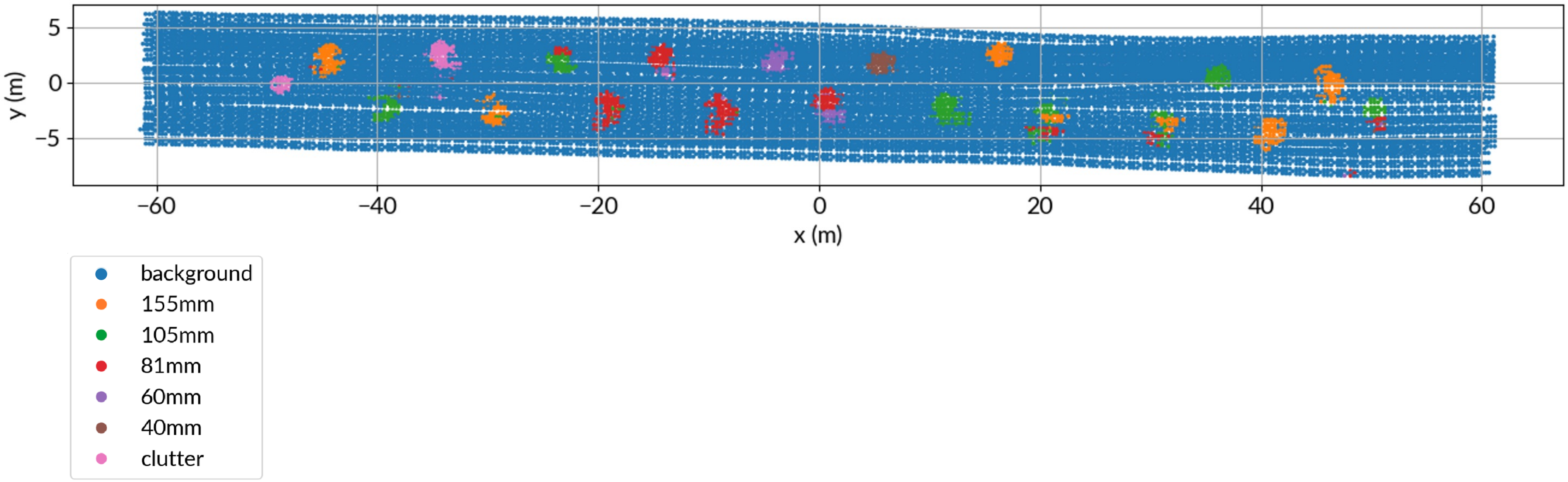
- 7 acquisition lines
- Current workflow requires seawater response removed
- Some ISOs present, we used only UXO objects to train (e.g. medium ISO ~ 81mm)

Get correlated noise using a binary classifier

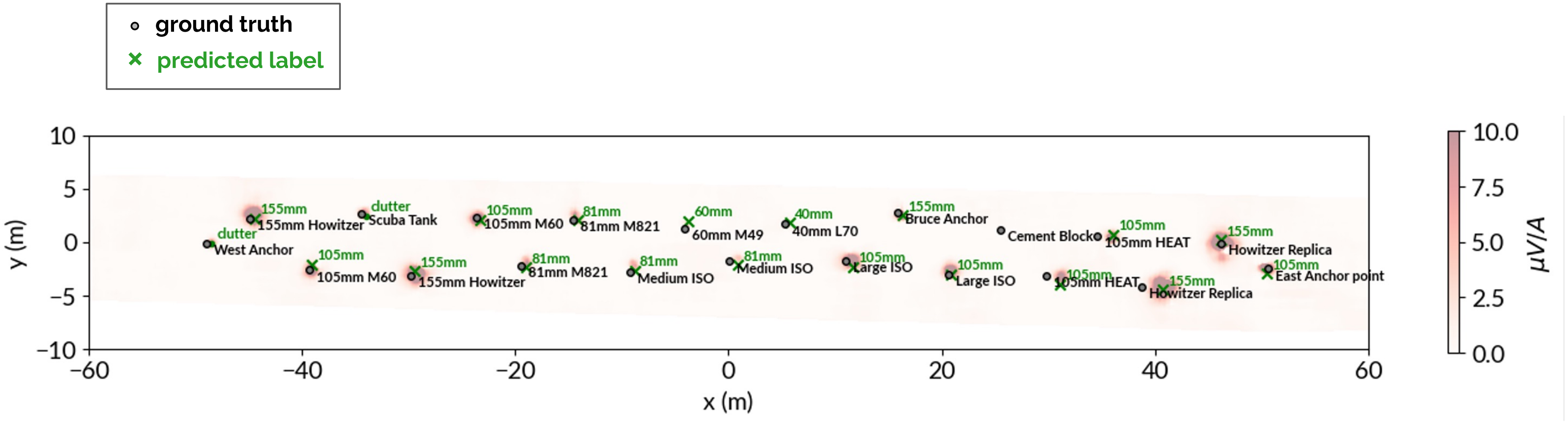


get spatially correlated noise from this subset of field data

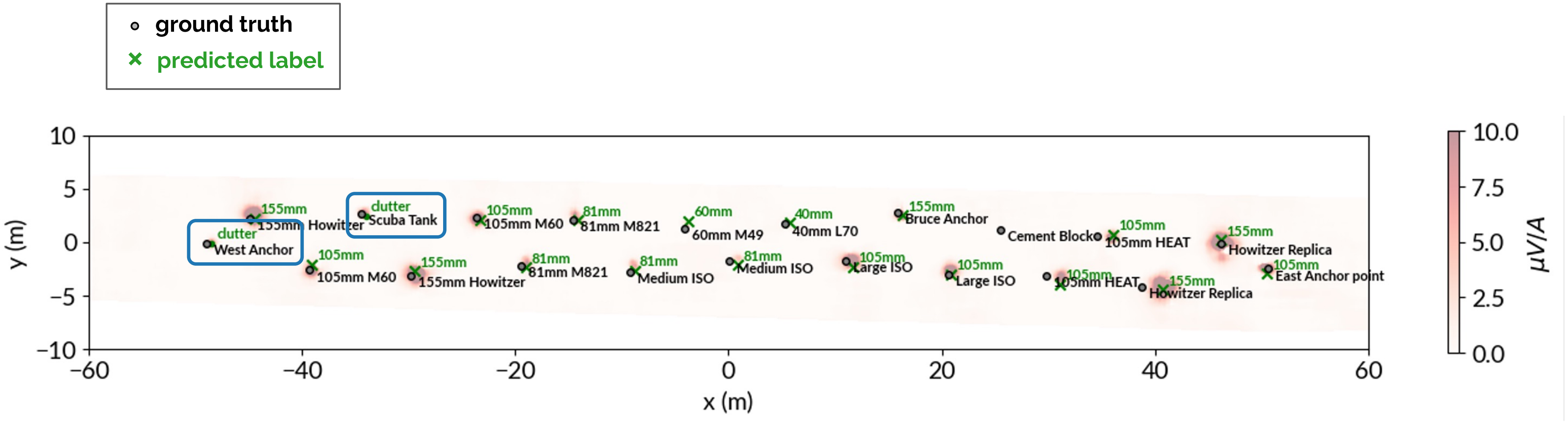
Classification map (output of CNN)



Predicted labels vs truth labels - field data

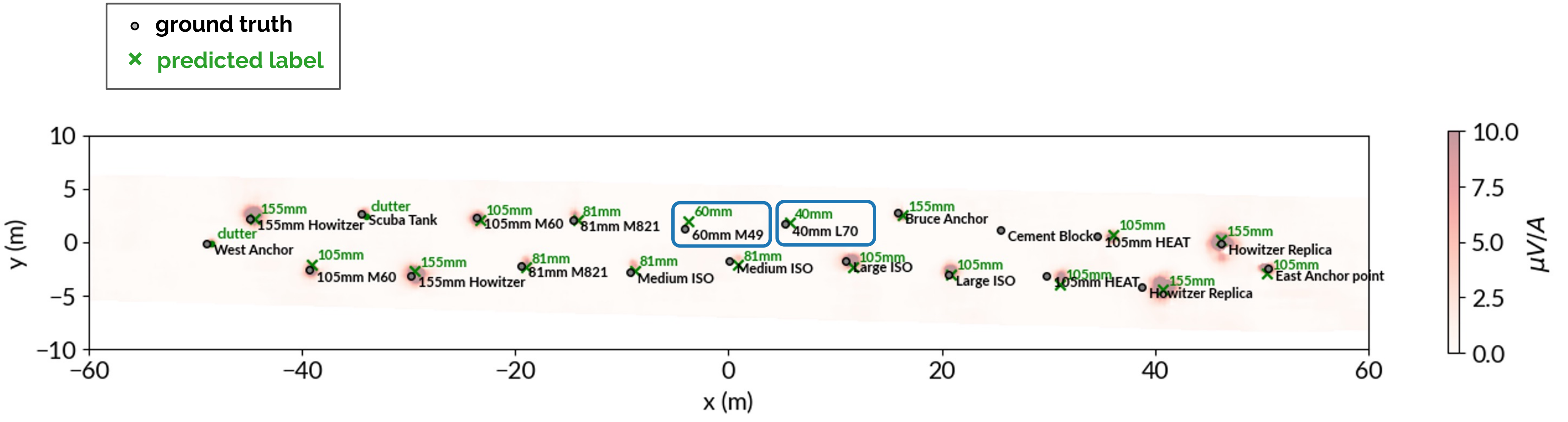


Predicted labels vs truth labels - field data



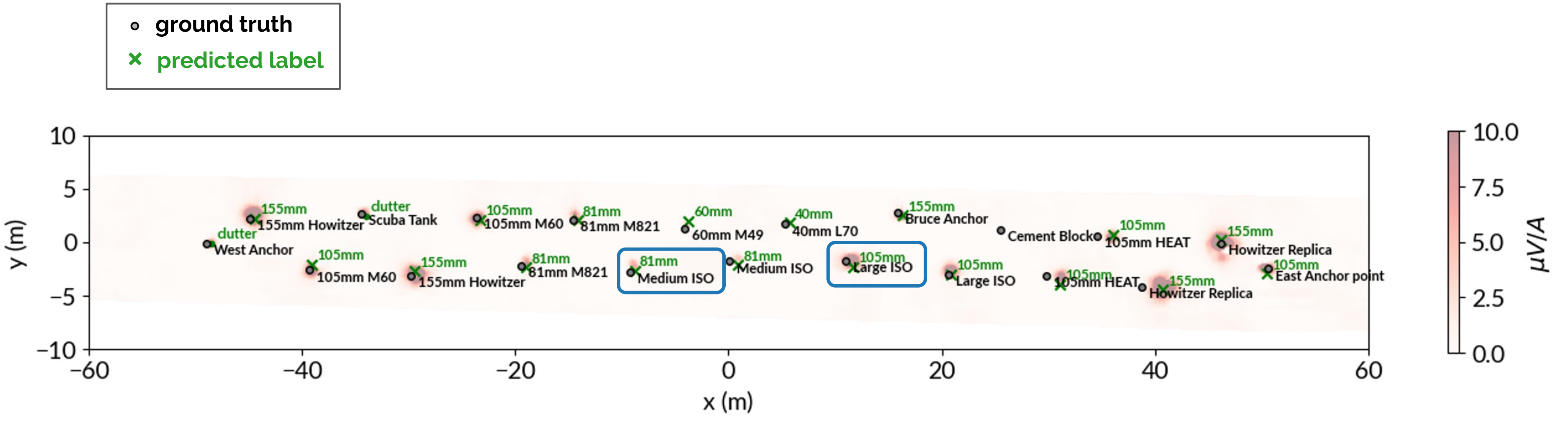
- Discriminated clutter

Predicted labels vs truth labels - field data



- Discriminated clutter
- Did not miss any UXO

Predicted labels vs truth labels - field data



- Discriminated clutter
- Did not miss any UXO
- Classified to closest object in training dataset

Concluding remarks:

- Key points:
  - image-segmentation architecture
  - clutter design and correlated noise are important
- Some limitations:
  - not trained to handle multiple objects in the same window
  - objects used to generate synthetic data should be close to the objects on the field
- Future work:
  - explore multi-target scenario (maybe instance segmentation)
  - combining with traditional approach

Concluding remarks:

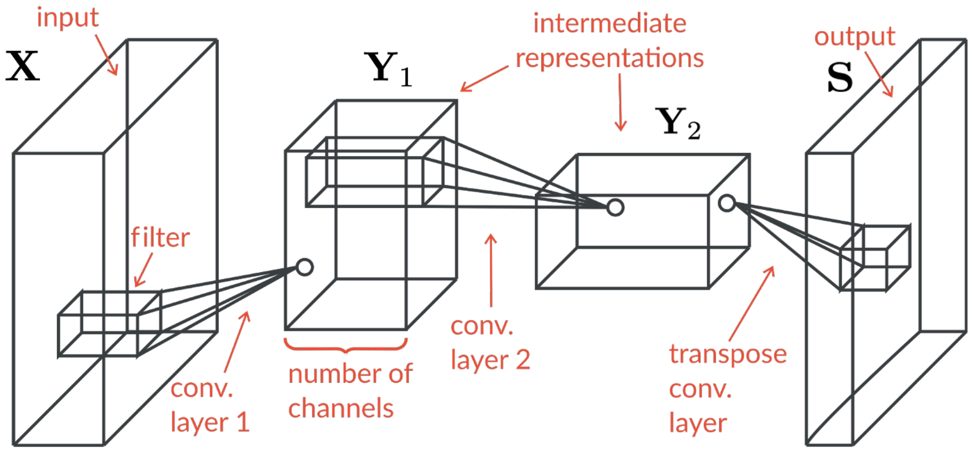
- Key points:
  - image-segmentation architecture
  - clutter design and correlated noise are important
- Some limitations:
  - not trained to handle multiple objects in the same window
  - objects used to generate synthetic data should be close to the objects on the field
- Future work:
  - explore multi-target scenario (maybe instance segmentation)
  - combining with traditional approach

Thank you!

Jorge Lopez-Alvis jlalvis@eoas.ubc.ca

Backup slides

Convolutional neural networks (CNNs)



$$\mathbf{s} = \mathcal{F}_\theta(\mathbf{X})$$

Mathematically:

$$\mathbf{Y}_1 = \sigma(\mathbf{K}_0\mathbf{X} + b_0)$$
$$\mathbf{Y}_2 = \sigma(\mathbf{K}_1\mathbf{Y}_1 + b_1)$$
$$\vdots$$
$$\mathbf{s} = \sigma(\mathbf{K}_{N-1}\mathbf{Y}_{N-1} + b_{N-1})$$

Convolutional Neural Networks (CNNs)

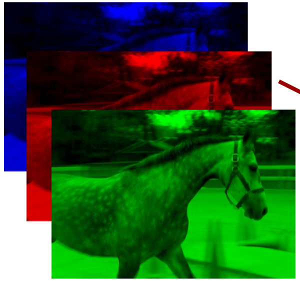
Training

define an optimization problem to estimate network parameters

Input

Features

$\mathbf{X}$



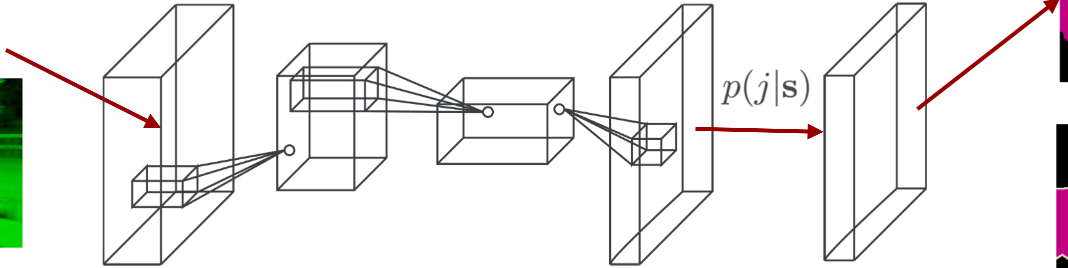
$(nx \times ny \times 3)$

Neural network

$$\mathbf{s} = \mathcal{F}_{\underline{\theta}}(\mathbf{X})$$

trainable parameters

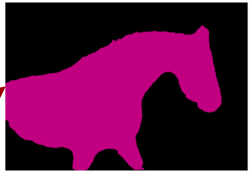
$\mathbf{S}$



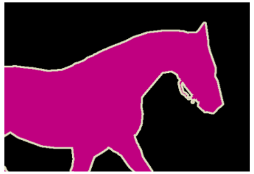
$$\mathbf{Y}_1 = \sigma(\underline{\mathbf{K}_0}\mathbf{X} + \underline{b_0})$$
$$\mathbf{Y}_2 = \sigma(\underline{\mathbf{K}_1}\mathbf{Y}_1 + \underline{b_1})$$
$$\vdots$$
$$\mathbf{s} = \sigma(\underline{\mathbf{K}_{N-1}}\mathbf{Y}_{N-1} + \underline{b_{N-1}})$$

Class probabilities

predicted



true

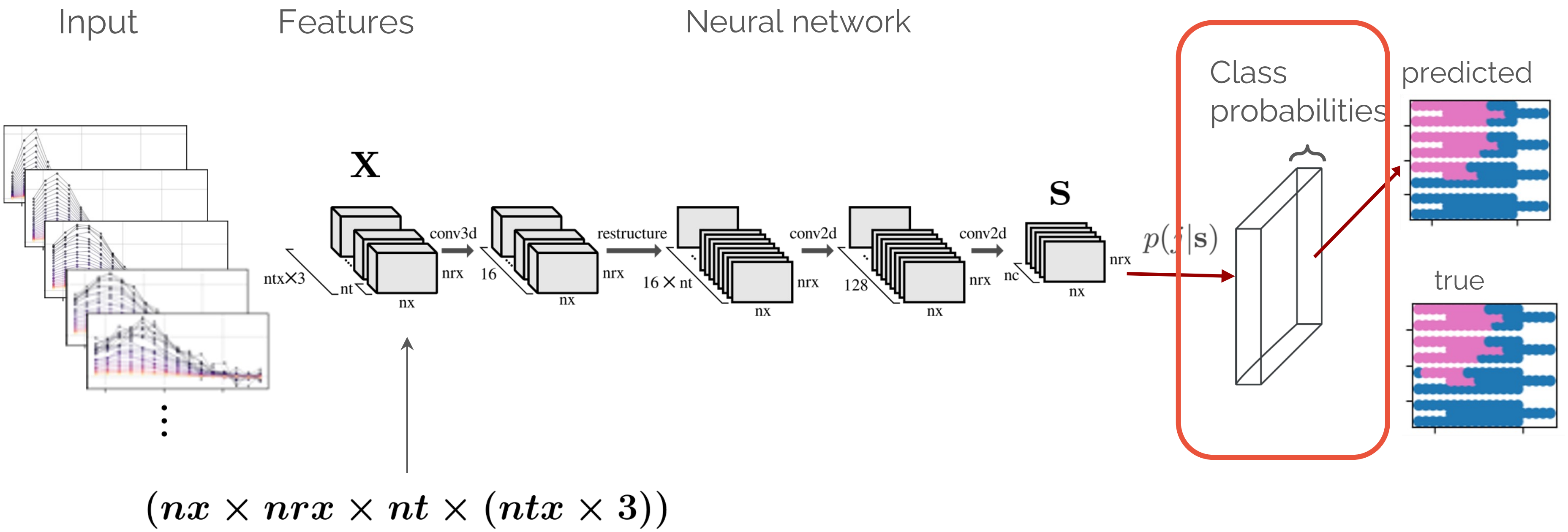


Measure: cross entropy loss

$$\min_{\theta} \phi = -\sum q_j \log(p_j)$$

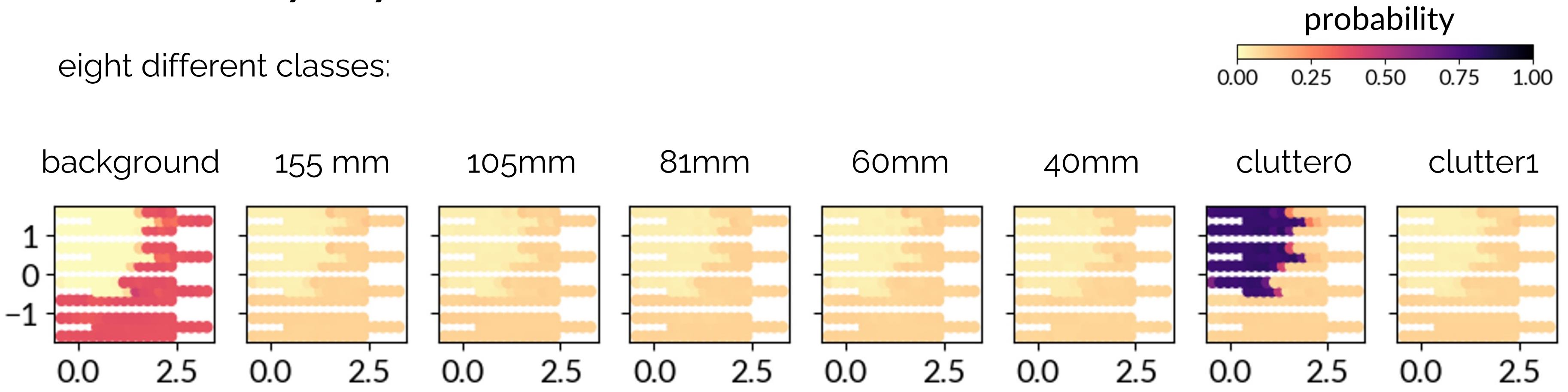
Convolutional Neural Networks

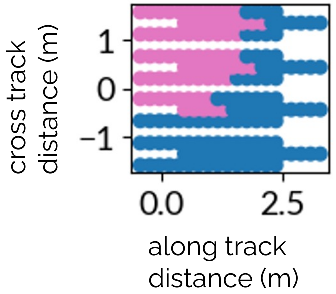
How do we translate these things to the UXO classification problem?



Probability layer and classification

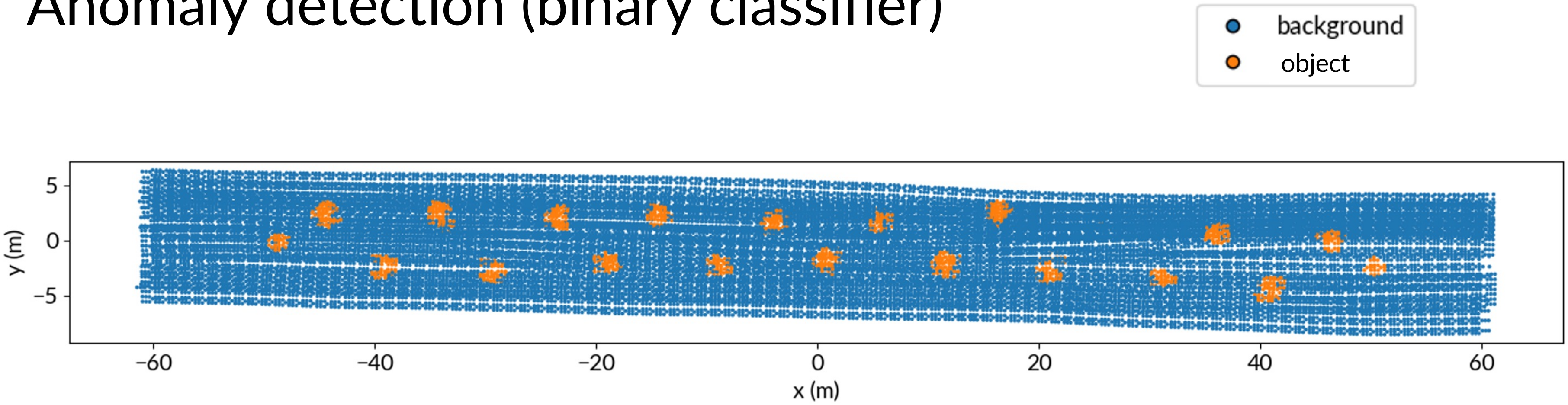
eight different classes:



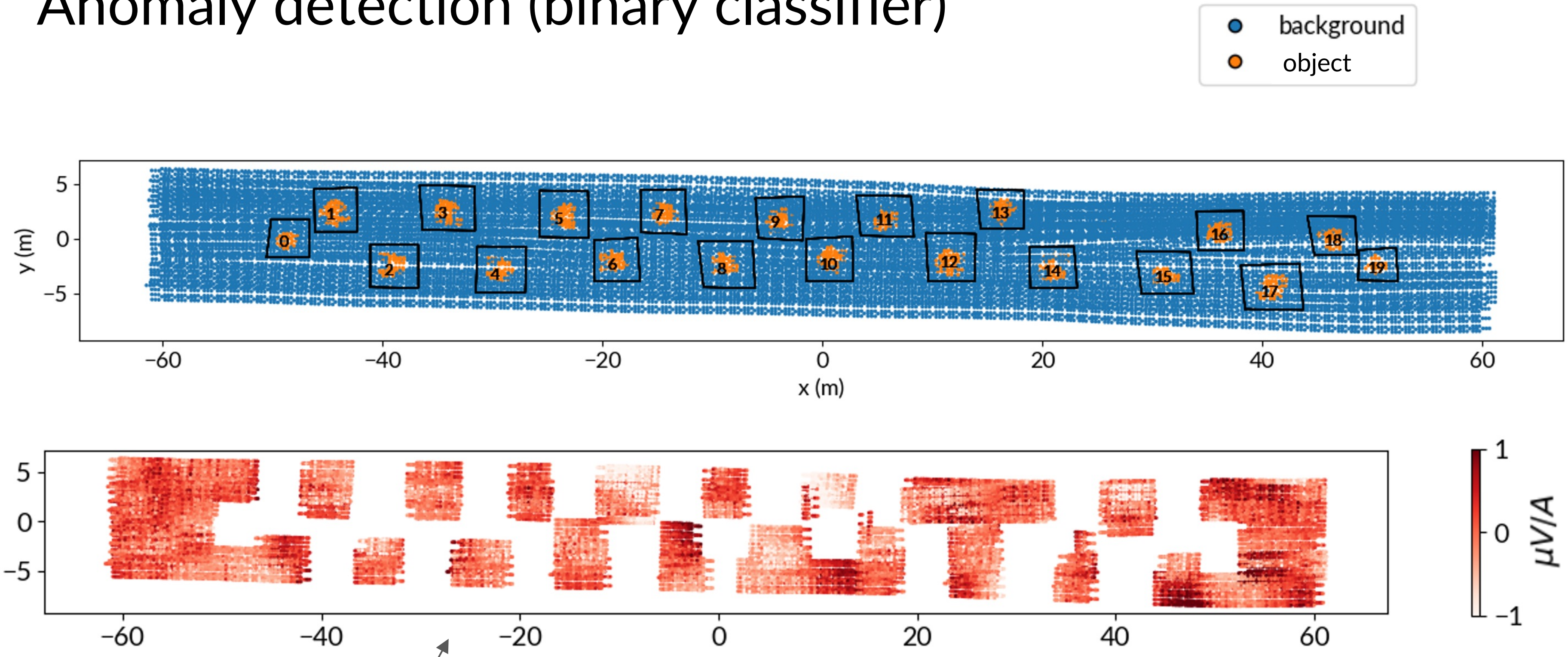


point-wise classification according to max probability

Anomaly detection (binary classifier)

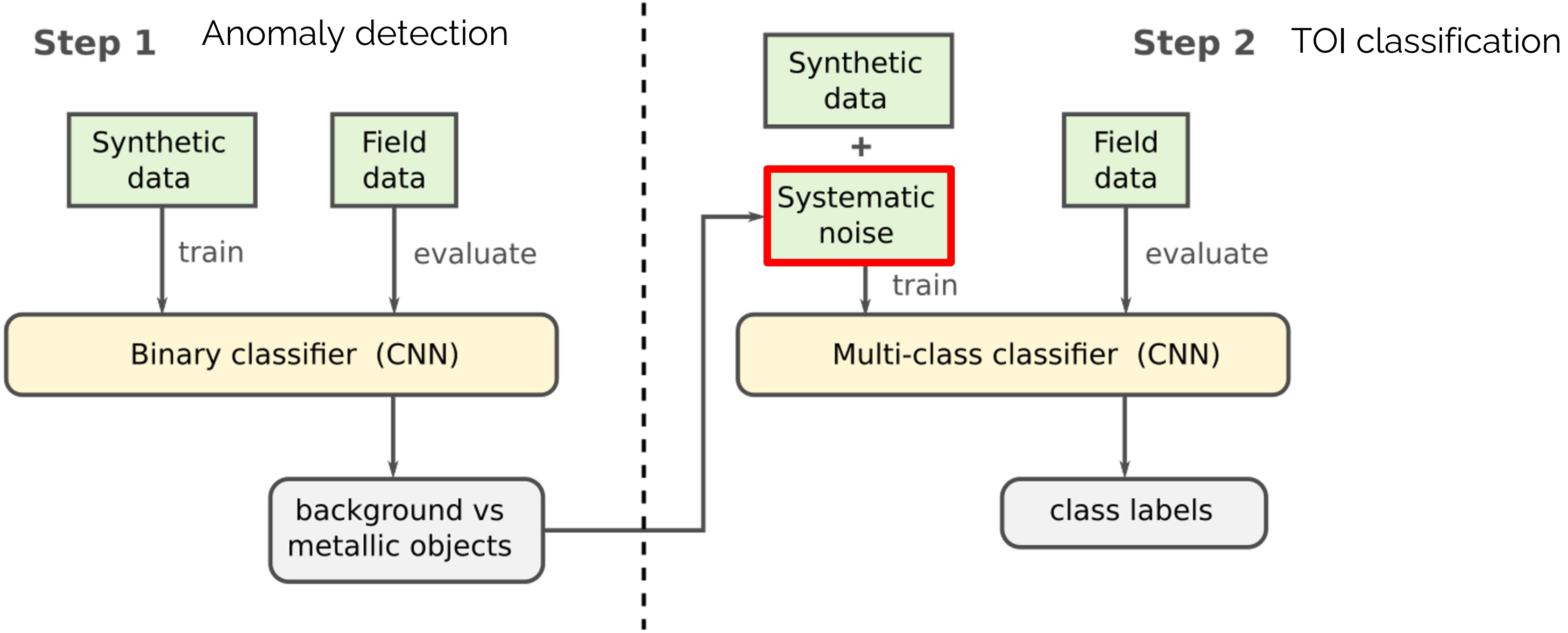


Anomaly detection (binary classifier)



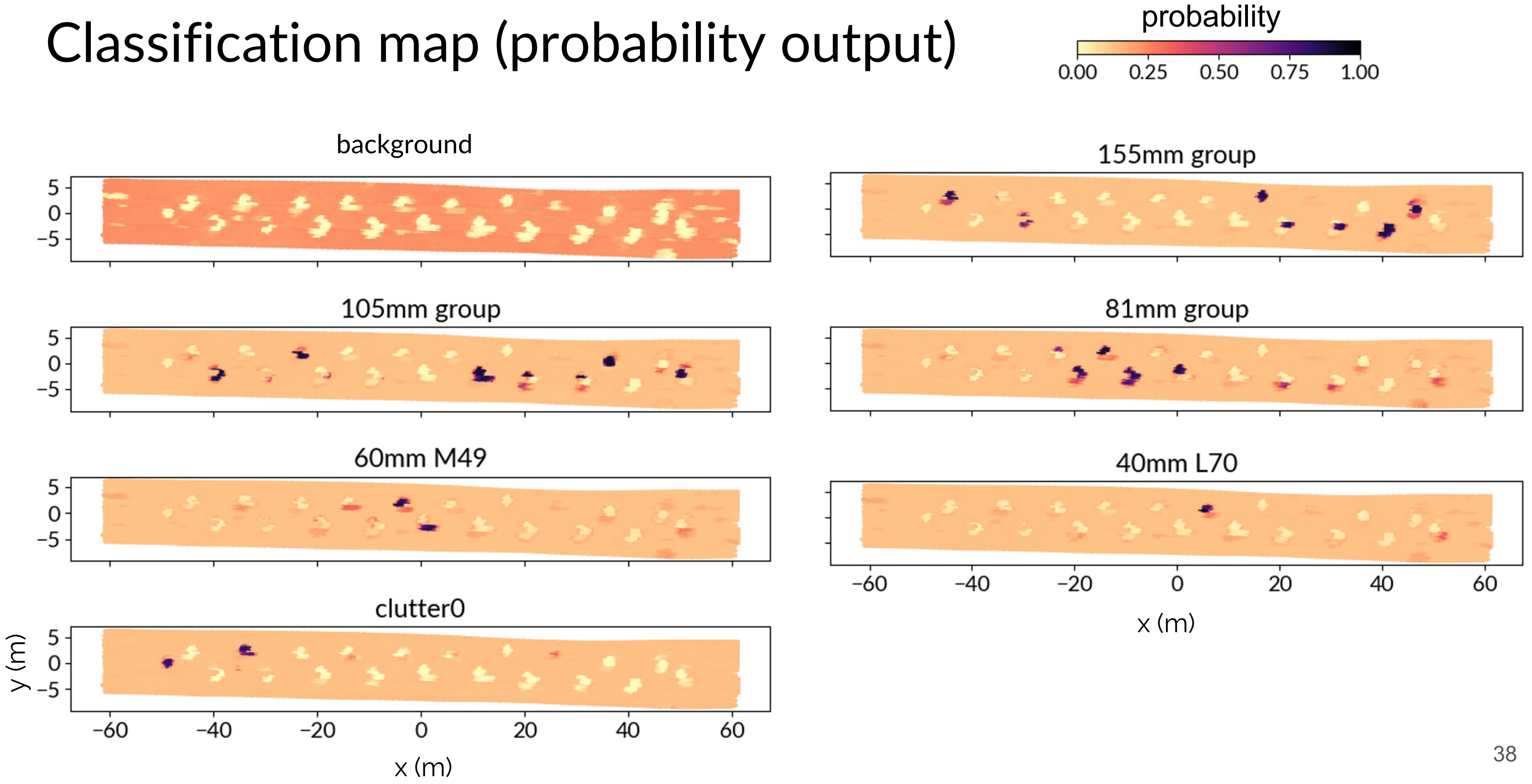
get spatially correlated noise from this subset of field data

Working with field data: two step workflow

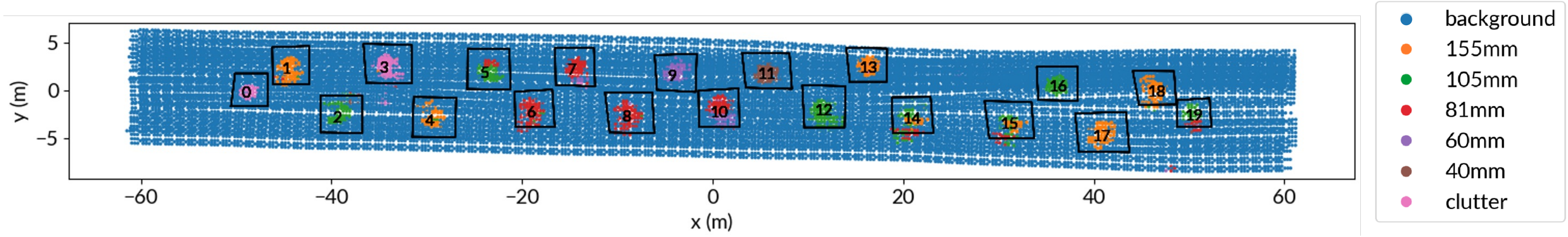


main goal: add realistic noise to the multi-class training dataset

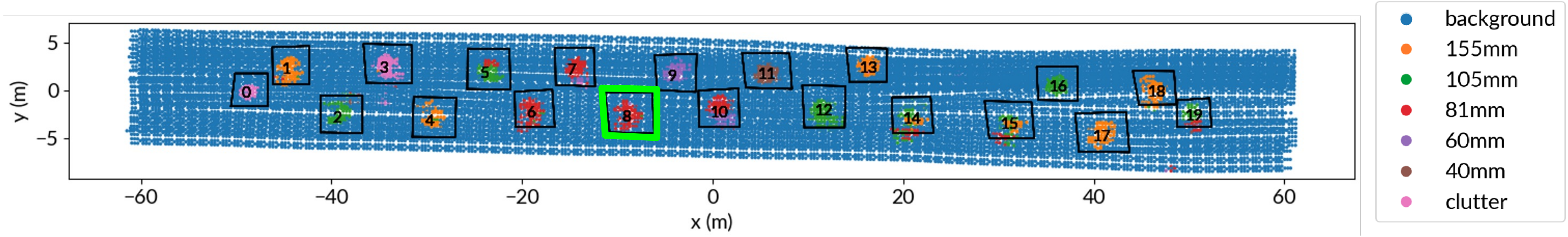
Classification map (probability output)



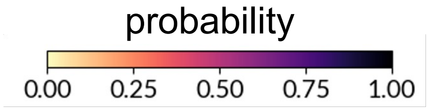
Divide in cells to get a single probability value per cell:

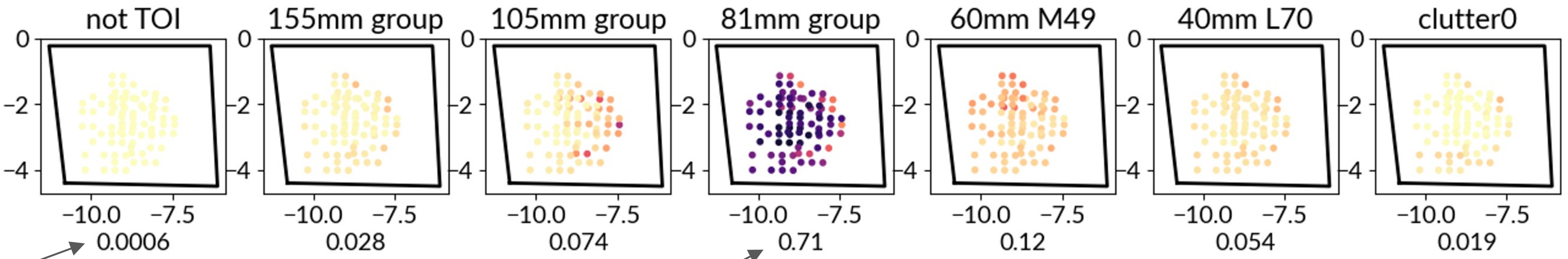


Divide in cells to get a single probability value per cell:



Get average probability for cell and assign final label

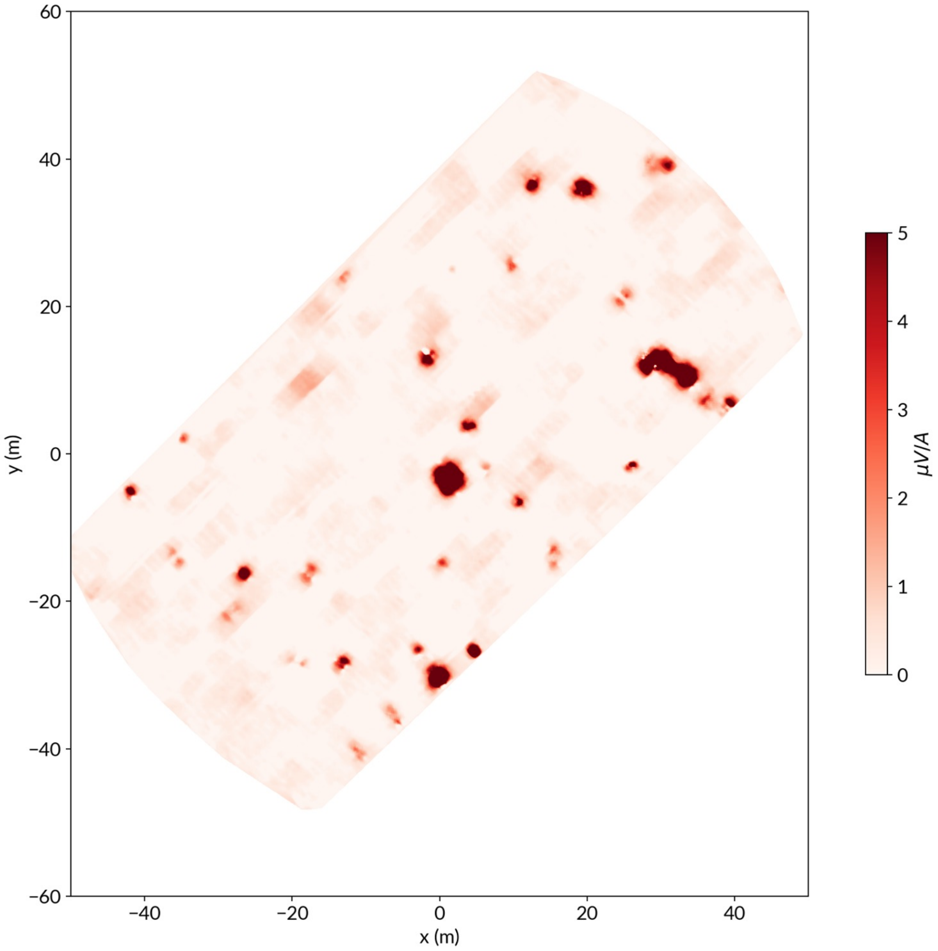




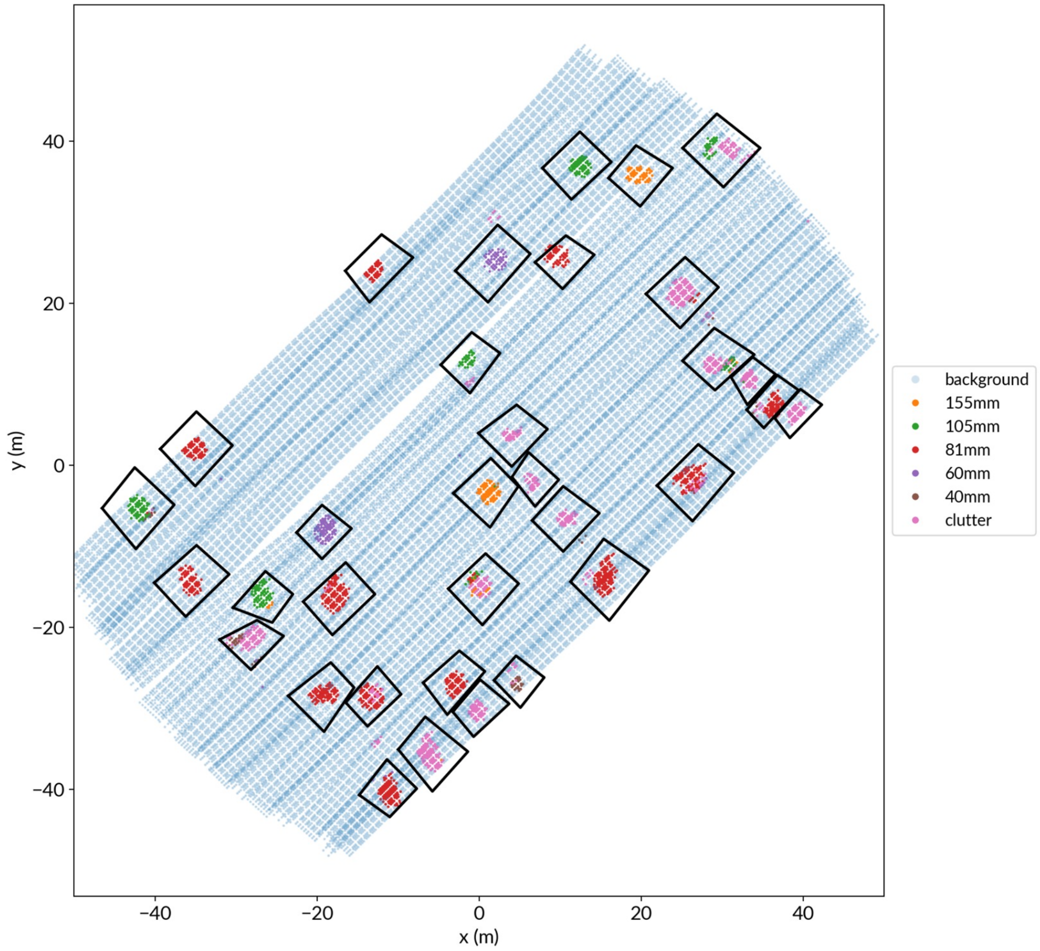
average probabilities

Assign label with highest probability: “81 mm”

Blindgrid 2021
Sequim Bay



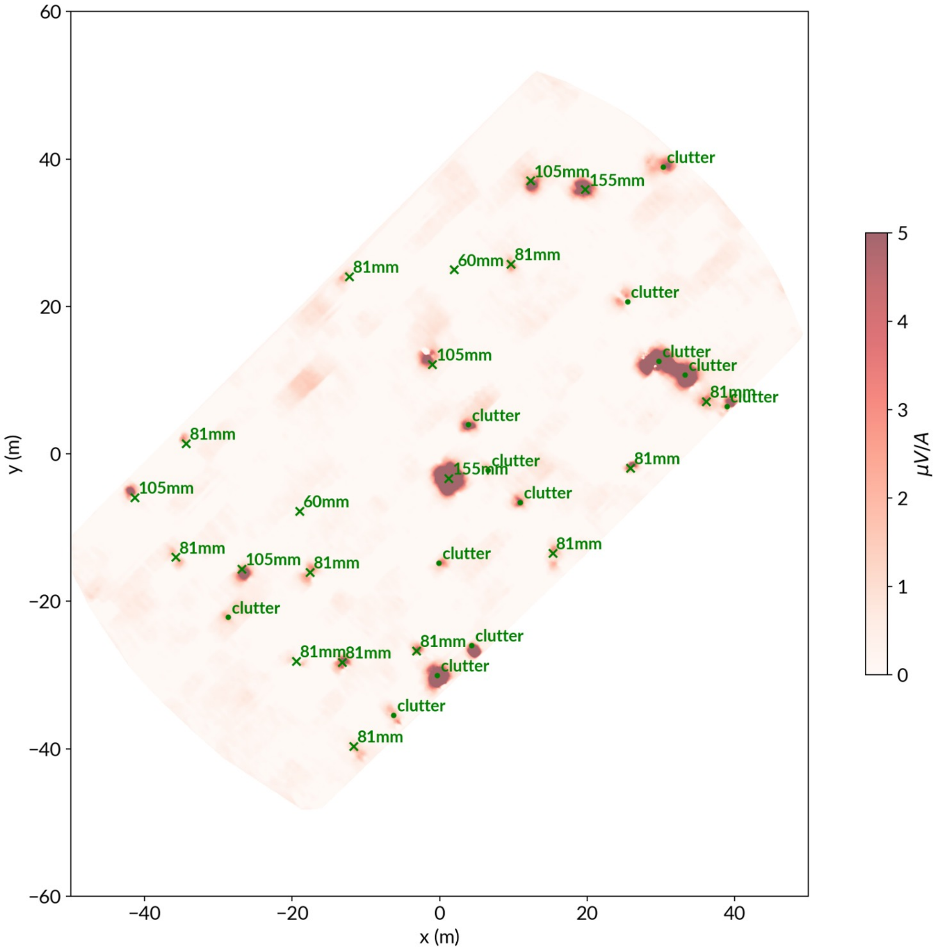
Blindgrid 2021
Sequim Bay



Blindgrid 2021
Sequim Bay

Predicted labels

- Missed only 1 UXO (out of 15)
- 11 out of 16 clutter labeled correctly

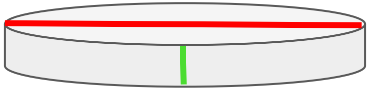


Clutter design

L1 and L2

L3

disk



PCA was helpful to decide whether clutter objects are very close to UXOs: