Machine Learning Models for Improved Tracking from Range-Doppler Map Images

Elizabeth Hou, Ross Greenwood, Piyush Kumar

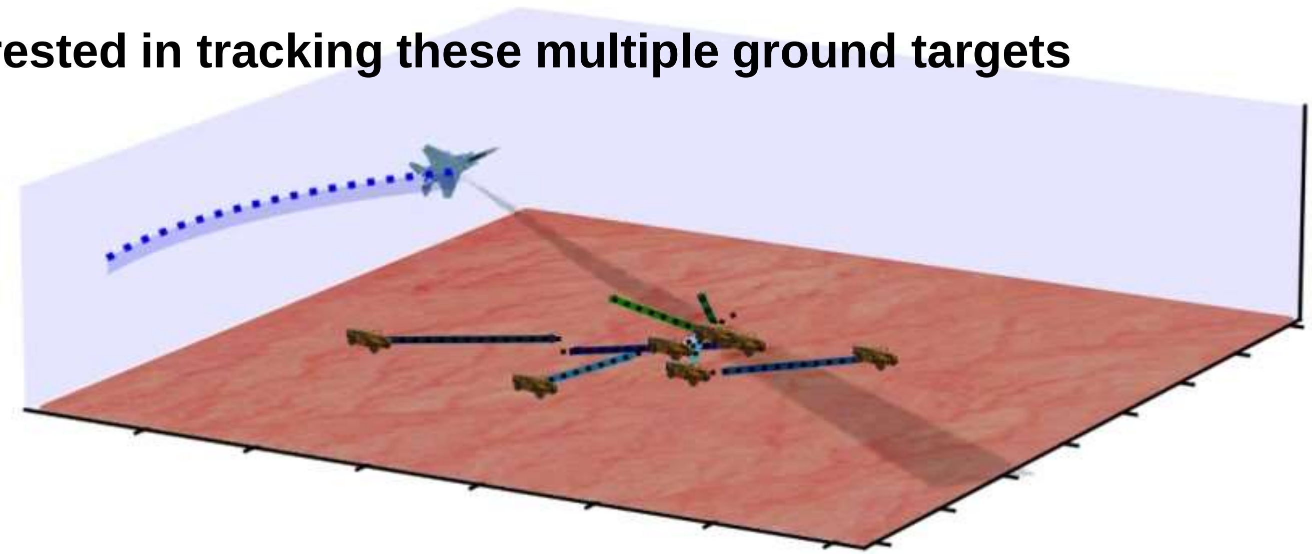
Systems & Technology Research

cyber ▪ analytics ▪ sensors ▪ systems

impact.

Problem Definition

- **A radar on an airborne platform is collecting measurements of targets on the ground**
- **The airborne platform's position is known in Cartesian coordinates affixed to the ground, i.e. East, North, Up (ENU)**
- **Moving targets on the ground create trajectories (latent state is position/velocity in Cartesian coordinates)**
  - For each target $\boldsymbol{k}$ there is a trajectory $\boldsymbol{z^k} = [\boldsymbol{z_1^k}, \ldots, \boldsymbol{z_t^k}]$
- **Each target's measurements $\boldsymbol{y}$ are its range, range-rate (Doppler), azimuth, and elevation angle relative to the airborne platform**
- **We are interested in tracking these multiple ground targets**



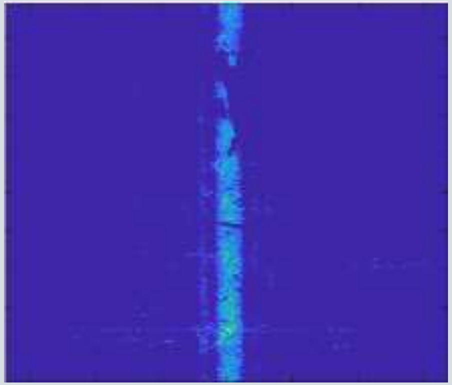
System Model

- **Dynamics Model:** $\boldsymbol{z_t} = \boldsymbol{\Phi z_{t-1}} + \boldsymbol{\omega_t}$
  - $\boldsymbol{z}$ *latent* state vector containing a target's positions and velocities (Cartesian coordinates)
  - $\boldsymbol{\Phi}$ *known* state transition matrix, describes targets movements between time points
  - $\boldsymbol{\omega}$ *known* process uncertainty (inherent noise in target's movements), distributed $\boldsymbol{N}(\boldsymbol{0}, \boldsymbol{Q})$

- **Measurement Model:** $\boldsymbol{y_t} = \boldsymbol{H z_t} + \boldsymbol{\varepsilon_t}$
  - $\boldsymbol{y}$ *observed* Doppler target vector *[range, range rate, azimuth, elevation]* (Spherical coordinates)
  - $\boldsymbol{H}$ *known* measurement matrix, converts from Cartesian space to Spherical space (assume linear)
  - $\boldsymbol{\varepsilon}$ measurement uncertainty (inherent noise from sensor e.g. thermal noise from machinery), distributed $\boldsymbol{N}(\boldsymbol{0}, \boldsymbol{R_t})$

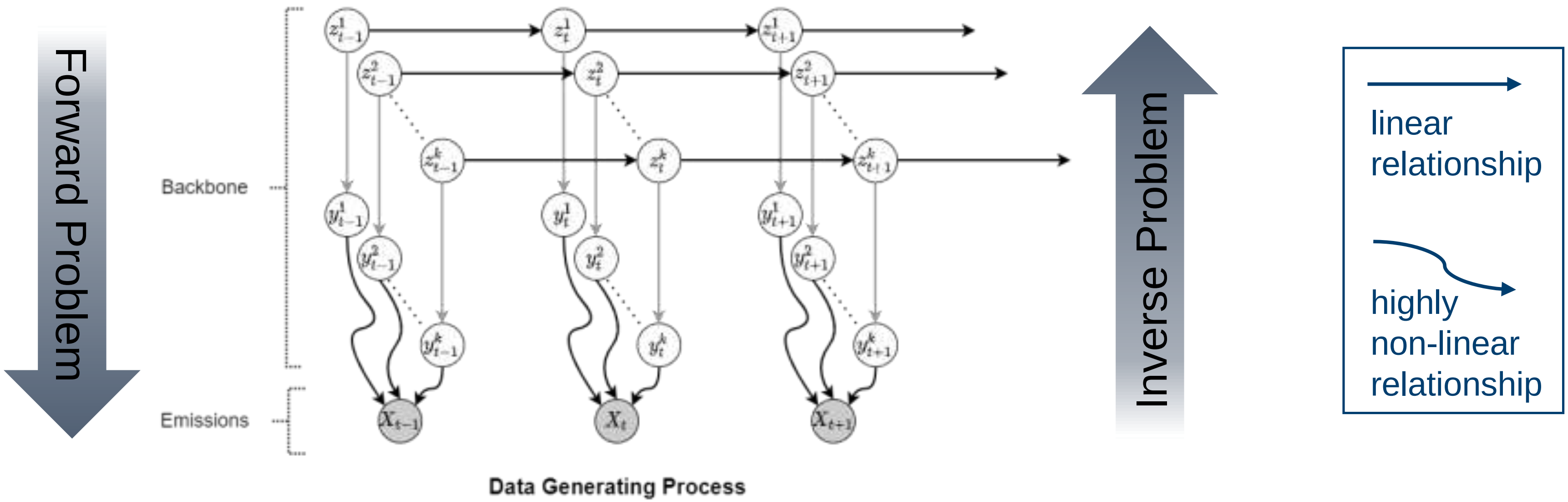
A radar / external sensor cannot directly measure $\boldsymbol{y}$!

- **Ground Moving Target Indication (GMTI) radars takes Range-Doppler Maps (RDM) images of these targets in their environment at various timepoints**
- **Each image $\boldsymbol{X_t}$ contains multiple target measurements $\boldsymbol{y_t^k}$. Image has "noise" present**



RDM images computed from measured IQ streams

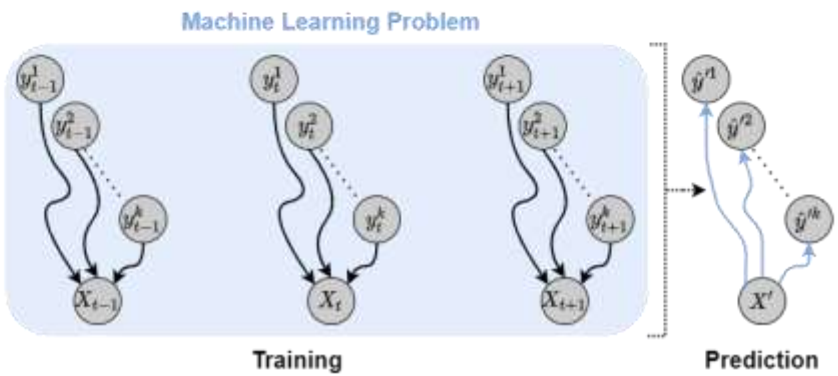
Inverse Problem: Detection + Tracking



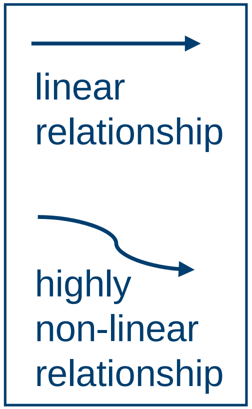
- **The data generating process (forward problem) generates observed data in the form of RDM images (emissions)**
- **Because the emissions are a *highly non-linear* function of the backbone, inverse problem is *hard***
- **Need to *learn* an inverse model for detecting target measurements!**

Supervised Target Detection

- **Observe some y_t^k labels containing *[range, range rate, azimuth, elevation]* for each very noisy image (RDM) X_t**
  - This noise distribution in X_t is highly complex due to the non-linearities even if the noise distribution in the latent space is additive Gaussian!

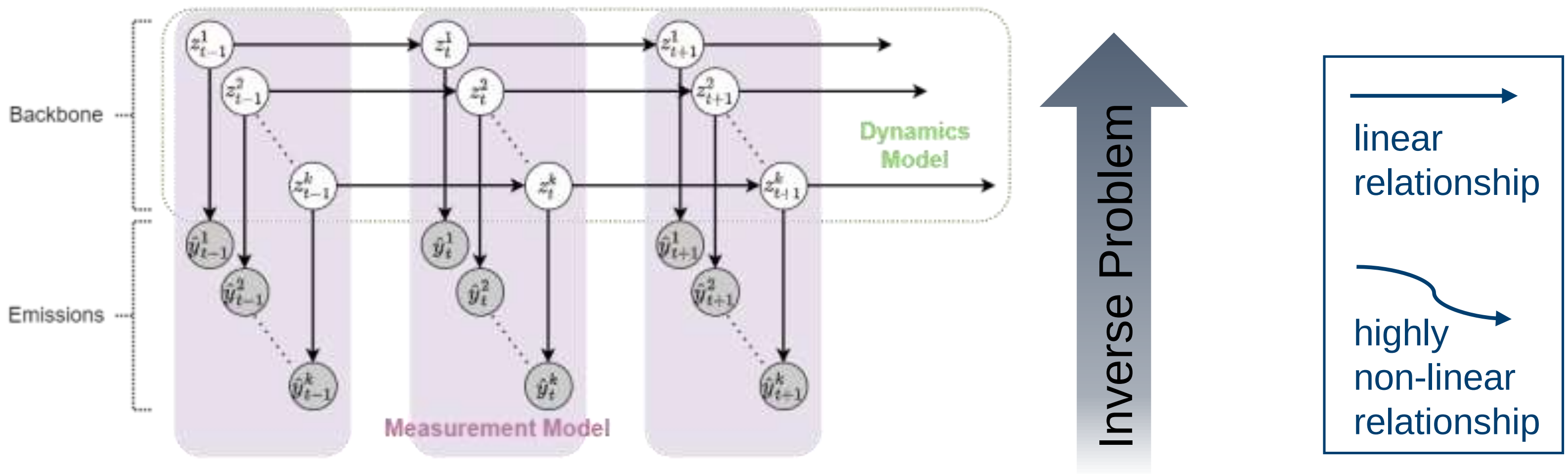


supervised
"Inverse Problem"



- **<u>Goal:</u> Train a Machine Learning (ML) model with labelled RDM images to predict new target measurements i.e. $\widehat{y_t^k}$ for all time points t and targets k when given new images**

Target Tracking as an Inverse Problem



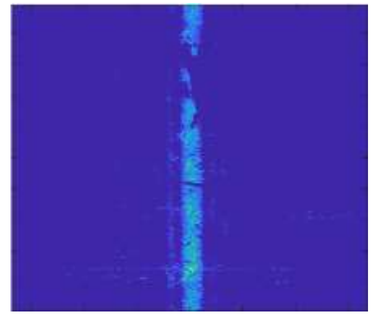
- **Replace emissions with ML model's predicted target locations $\widehat{y_t^k}$**
- **All relationships are now *linear,* so inverse problem is now much more *tractable***
  - All Kalman filter based tracking models are useable provided we know measurement uncertainty

Proposed Solution

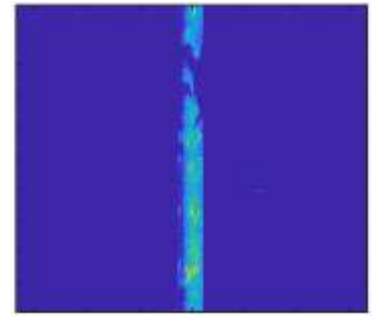
Sensor Model

- Need large amounts of RDM image, target location pairs $\{X_n, Y_n\}$ for labelled training data
- Sim model injects targets into simulated RDMs using the radar parameters corresponding with a real image and physics-based equations

Real



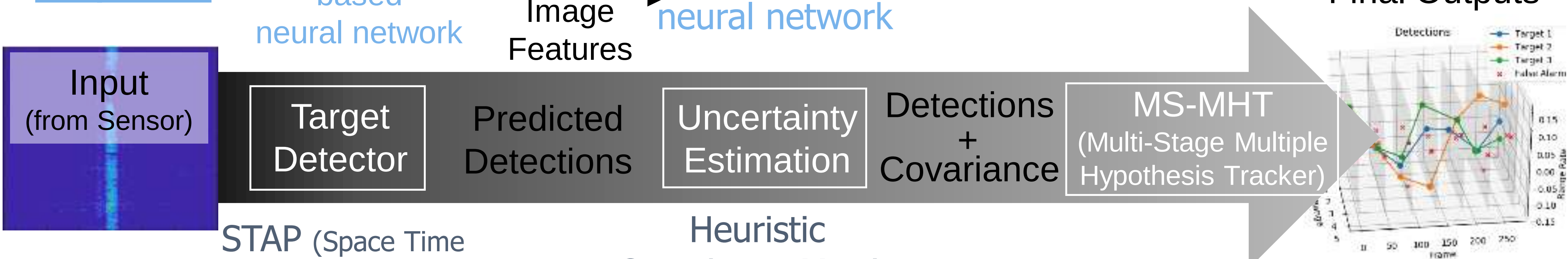
Generated



Estimation Task

Proposed: Calibrated UNet based neural network - - Extracted Image Features - - → CVAE based neural network

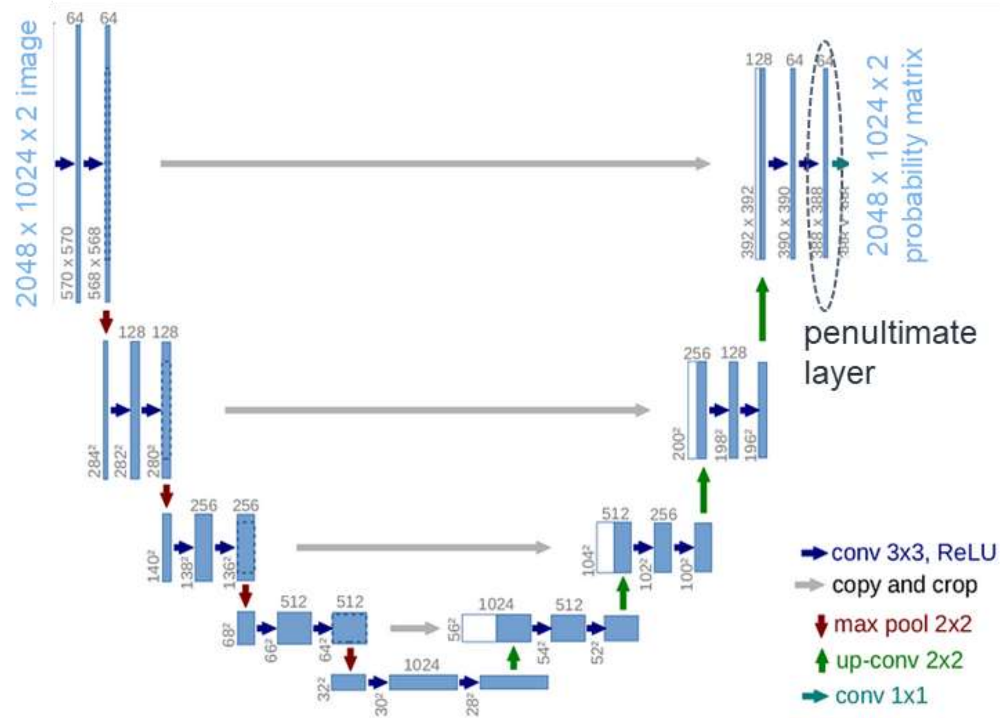
Final Outputs



Baseline: STAP (Space Time Adaptive Processing) + CFAR (Constant False Alarm Rate) — Heuristic Covariance Matrix (using on radar parameters)

Target Detection Model

- **Discriminative method with a UNet architecture trained on labeled data**
  - $\boldsymbol{X_n}$ is a $h\ x\ w\ x\ m$ complex matrix where h and w are pixel dimensions and m = number of channels in a radar
  - $\boldsymbol{Y_n}$ is a $h\ x\ w$ binary matrix indicating whether each pixel contains a target or not
  - For each pixel it *learns* features (containing info from other pixels) to predict if there's a target
- ***Learns* to ignore endo-clutter noise / *learns* STAP**

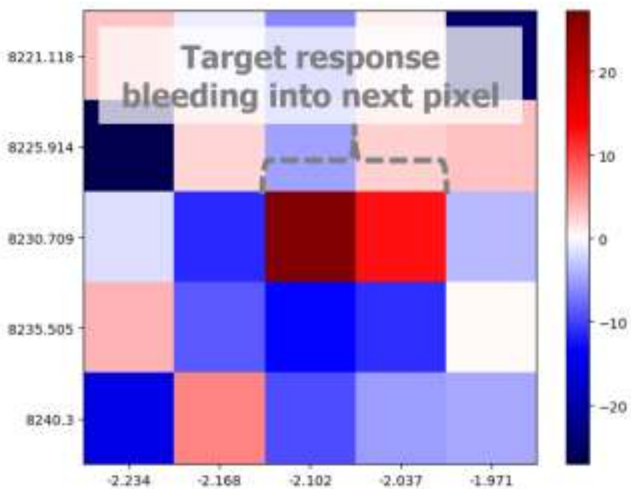


$$\mathcal{L}(Y,\hat{Y}) = -\sum_{i,j} w^1 Y_{(i,j)} \log \hat{Y}_{(i,j)} + w^0 (1 - Y_{(i,j)}) \log(1 - \hat{Y}_{(i,j)})$$

weights for class imbalance

Discrete to Continuous Measurements

- **Pixel level classification gives discrete bins of target locations**
  - Threshold and assign each predicted measurement $\hat{y}^k$ to be the corresponding range and range-rate bins of the pixel
- **RDM images are capturing aspects of a "continuous" real world in discrete sensor measurements,**
  - Targets may not fall exactly within a pixel bin and instead between pixels
- **Want one measurement per target for our Filter's measurement model**



- **Use weighted averaging**

$$\hat{y}^k = \frac{\sum_{(i,j)\in W(k)} s_{(i,j)}\,\hat{Y}_{(i,j)}}{\sum_{(i,j)\in W(k)} \hat{Y}_{(i,j)}}$$

Range and rate-rate coordinates for pixel (i, j)

Post softmax probabilities (weights) for pixel (i, j)

Patch of pixels centered around predicted target k

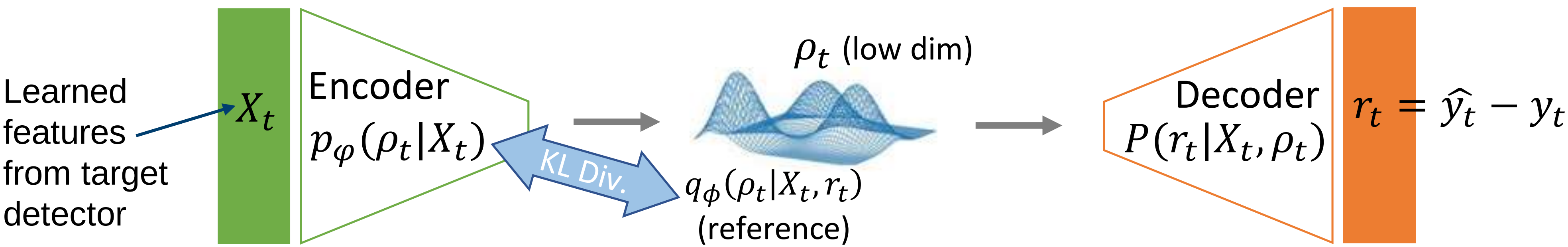
Statistical Model of the Target Detector

- **Also need measurement covariances for Filter's measurement model**
- **Use Conditional Variational Auto-encoder (CVAE) to learn their distribution**

$$\underbrace{\max_{\Theta,\phi,\varphi}}_{\text{distributions' parameters}} \mathcal{L}_{CVAE} = -KL\left(q_\phi(\rho_t|X_t, r_t)||p_\varphi(\rho_t|X_t)\right) + E_{\rho\sim q_\phi(\rho_t|X_t,r_t)}\left(\log P(r_t|X_t,\rho_t)\right)$$

distributions' parameters

- **Pixels in endo-clutter region also have noise from the ground clutter returns**
  - Twin CVAE architecture with separate distributions for end and exo-clutter regions

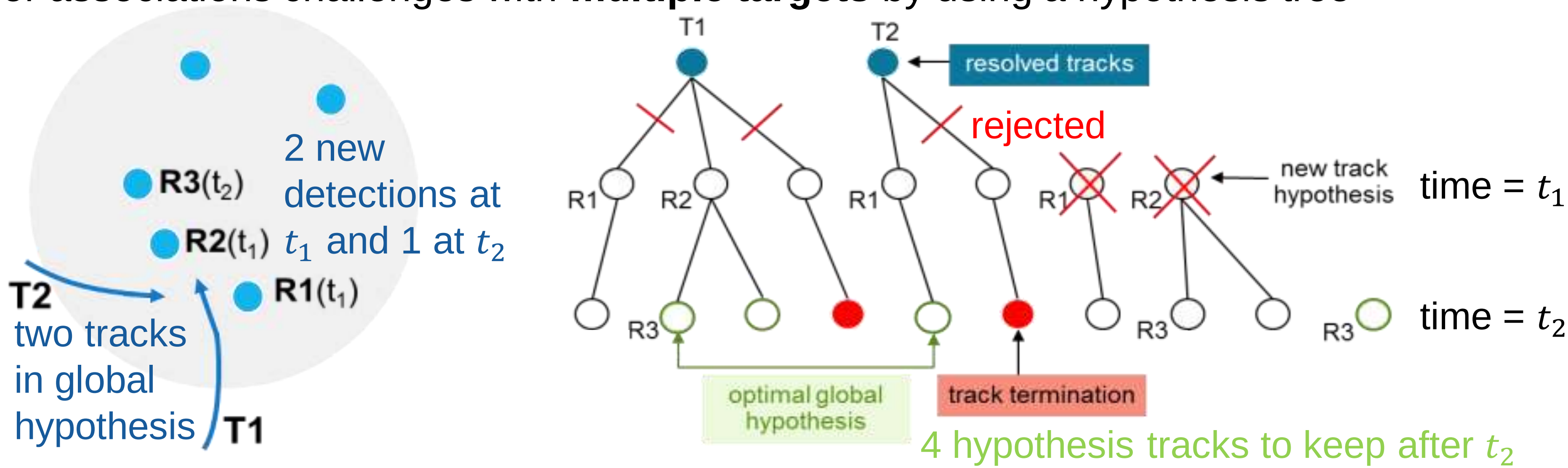


- **Trained Encoder-Decoder form a Mixture Model to sample from given a new input image to empirically calculate $\Sigma(y'_t|X'_t)$**

$$P(r_t|X'_t) = \int P(r_t|X'_t, \rho_t)\, p_\varphi(\rho_t|X'_t)\, d\rho_t$$

Statistical Tracker

- UNet model provides **target detections** in terms of estimated *[range, range rate, azimuth, elevation]* for use as the "measurements" in a statistical filter's measurement model
- CVAE models provides **measurement noise estimates** of the original noise covariance in the Spherical coordinates of the statistical filter's measurement model
- A (standard) Statistical Filter (given a dynamics model) estimates the latent state of the targets' positions and velocities in 3D (Cartesian coordinates – East, North, Up)
- The Multi-Stage Multiple Hypothesis Tracking Algorithm extends standard filtering to account for associations challenges with **multiple targets** by using a hypothesis tree



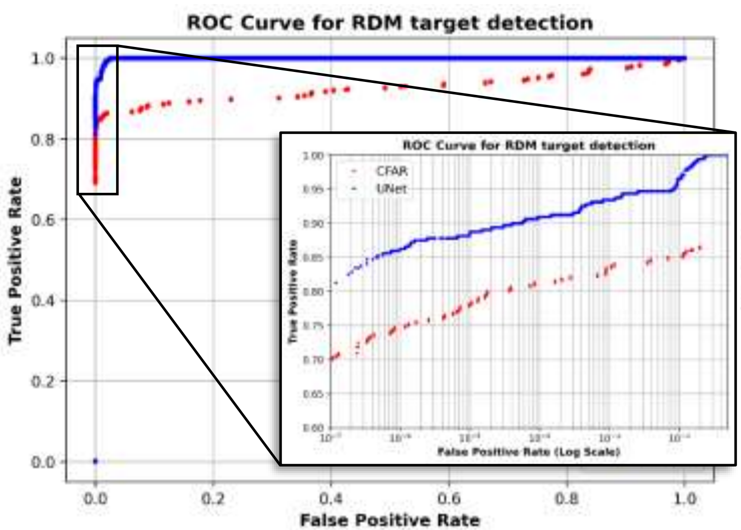
Target Detector Accuracy

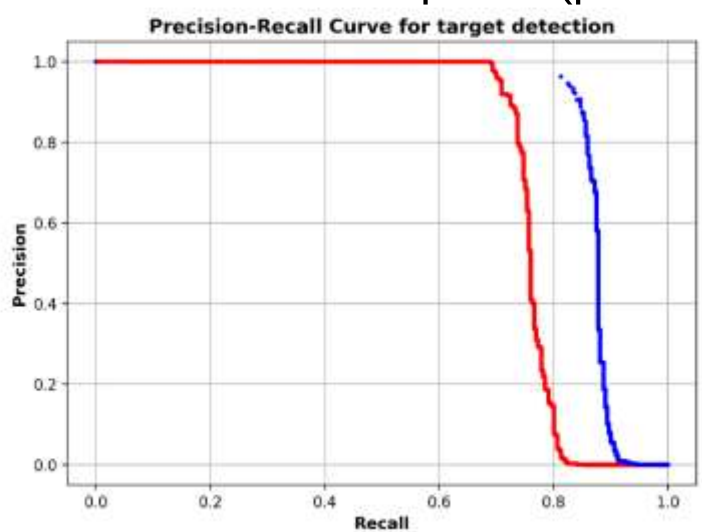
Existing baseline processing technique: STAP + CFAR

- STAP: (Space-time Adaptive Processing) whitens / removes the clutter noise
- CFAR: (Constant False Alarm Rate) Neyman-Pearson statistical hypothesis test for each pixel (Uniformly most powerful test but only when distribution is *correctly specified* **(not true here)**)

UNet Target Detector

- UNet based neural network: Train a discriminative model using labelled data to learn
  1. To ignore the clutter noise in the endo-clutter region
  2. Features that contain information from other pixels (pixels not treated independently)





At all false positive rates, UNet *statistically* more powerful

	TPR	FPR
CFAR	0.75	10^{-6}
UNet	0.86	10^{-6}

Approximately 2.1 false detections per image

Improving Tracking Accuracy

- **Baseline-Filter:**
  - STAP for pre-processing, Constant False Alarm Rate (CFAR) model for target detection, a constant covariance matrix, and MHT for target tracking
- **ML-Filter:**
  - Trained UNet based neural network for target detection, Trained CVAE based neural network for uncertainty estimation, and MHT for target tracking

Metric	Constant Velocity (Simple)		MoveStopMove (Complex)	
	ML-Filter	Baseline-Filter	ML-Filter	Baseline-Filter
TaC	0.5075	0.5127	0.9482	0.9848
TrC	1	1	1	0.6476
TaP	0.9639	0.9428	0.9050	0.8756
TrP	1	0.995	0.9816	0.9685
LE	0.007	0.114	0.0101	0.1403

Higher is better (TaC, TrC, TaP, TrP)

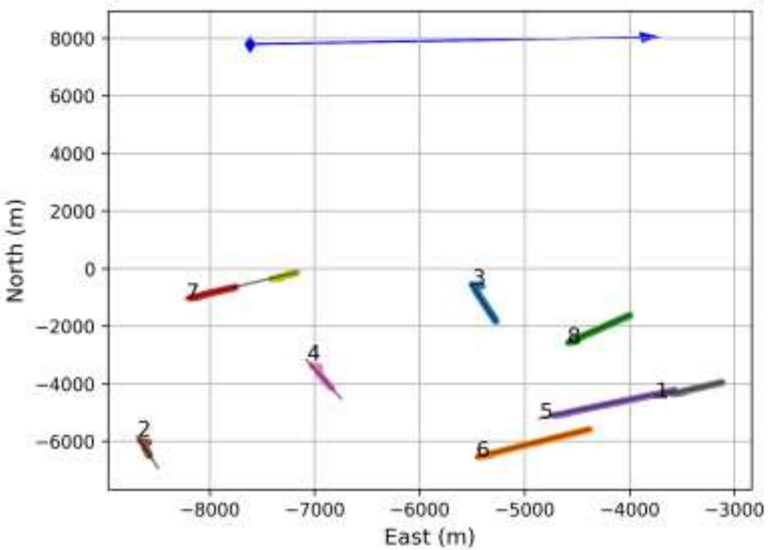
Lower is better (LE)

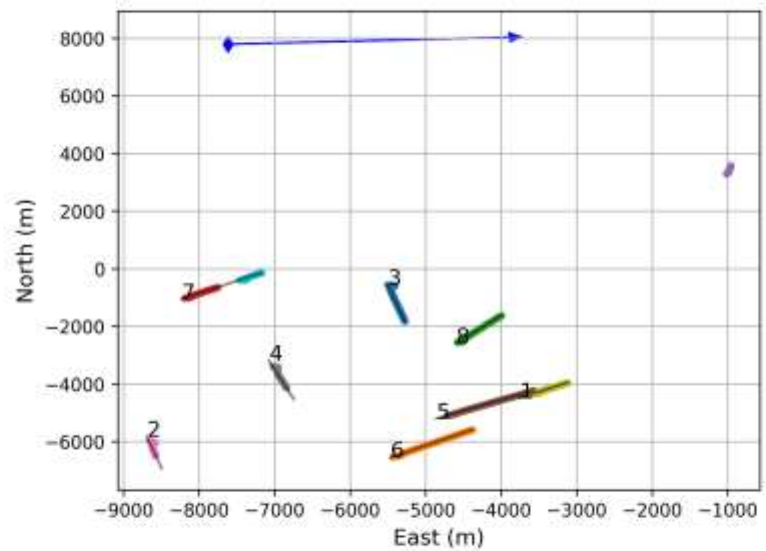
Comparable in Simple Scenario, slightly worse in target completeness (TaC), but significantly better in track completeness (TrC) in Complex Scenario

Improving Tracking

Baseline-Filter

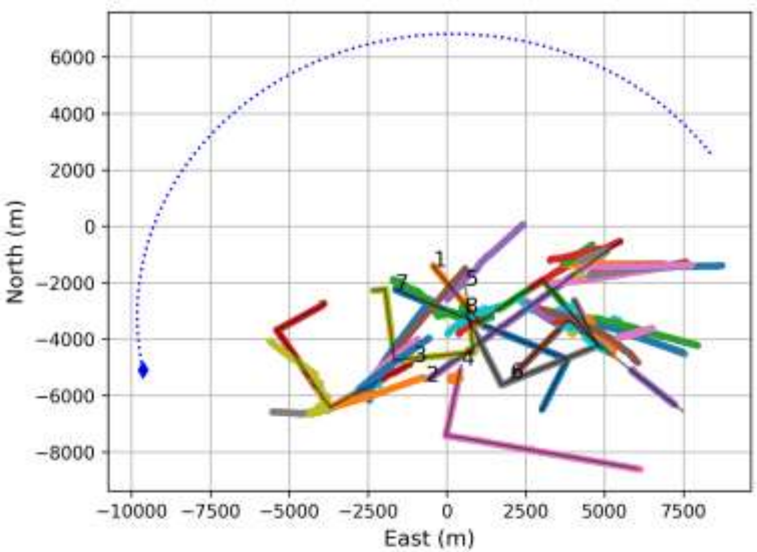
Simple Scenario

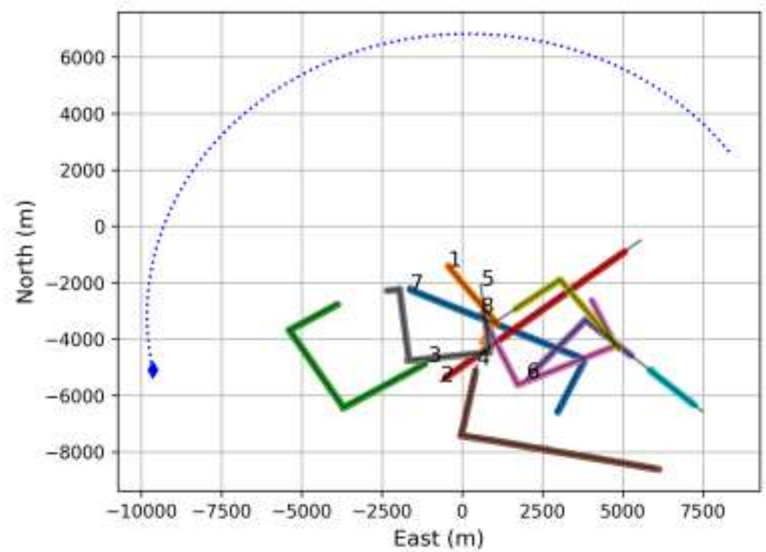




ML-Filter

Complex Scenario





Thank You

Questions?