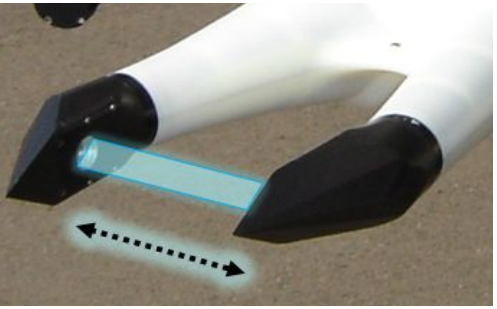
Holodec Hackathon

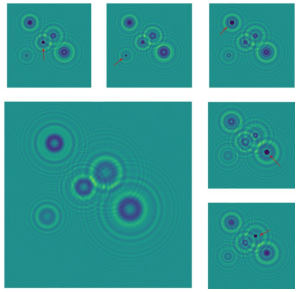
HOLODEC Machine Learning Challenge Problem

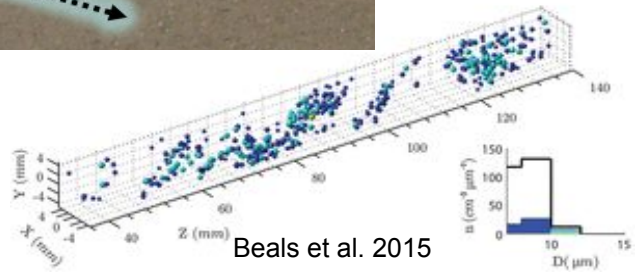
Matt Hayman, Aaron Bansemer, David John Gagne, Gabrielle Gantos, Gunther Wallach, Natasha Flyer

Holographic Detector for Clouds (HOLODEC)

- Airborne instrument that measures liquid droplets and ice crystals in natural clouds
- Simultaneously measures all particles in 13 cm^3 volume leading to 1000+ particles per hologram
- Refocusing performed on individual particles to return the 3-D position of the particle (x, y, z) as well as the size (d) and shape (2 million core hours per project)



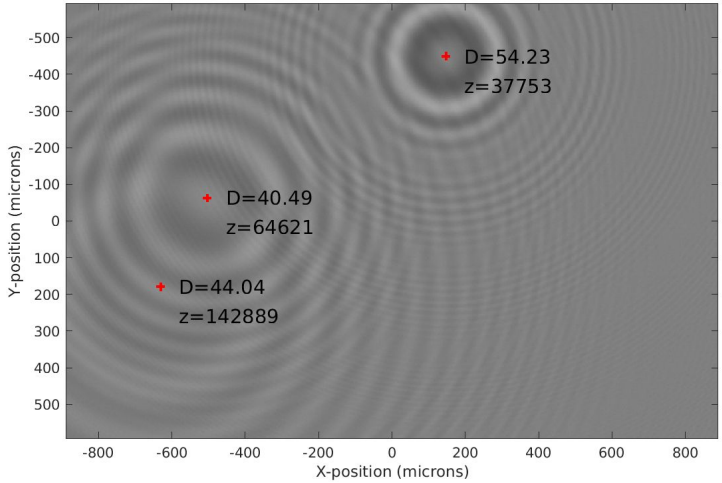




Beals et al. 2015

Can ML reduce core hours?

- Synthetic data using simplified holograms
  - Circular droplets only (no ice)
  - Synthetic grayscale hologram images
  - X,Y,Z-position and Diameter of each particle
- Training (15,000) and validation sets for 1 and 3 particles per 2-D hologram image
- Goal: learn X,Y,Z-position and D of each particle



Baseline Models

Single Particle

- 2D CNN model attempting to learn per-hologram particle x,y,z position and diameter

Layer (type)	Output Shape	Param #
input (InputLayer)	[(None, 600, 400, 1)]	0
conv2D_00 (Conv2D)	(None, 600, 400, 16)	416
maxpool2D_00 (MaxPooling2D)	(None, 150, 100, 16)	0
conv2D_01 (Conv2D)	(None, 150, 100, 24)	9624
maxpool2D_01 (MaxPooling2D)	(None, 37, 25, 24)	0
conv2D_02 (Conv2D)	(None, 37, 25, 32)	19232
maxpool2D_02 (MaxPooling2D)	(None, 9, 6, 32)	0
flatten (Flatten)	(None, 1728)	0
dense_00 (Dense)	(None, 64)	110656
dense_01 (Dense)	(None, 32)	2080
dense_output (Dense)	(None, 4)	132

Total params: 142,140
Trainable params: 142,140
Non-trainable params: 0

- Mean Absolute Error

Variable Name	Error
x	290 µm
y	170 µm
z	53,271 µm
d	16 µm

Three Particle

- 2D CNN model attempting to learn per-hologram particle mass distribution along the z-axis

Layer (type)	Output Shape	Param #
input (InputLayer)	[(None, 600, 400, 1)]	0
conv2D_00 (Conv2D)	(None, 600, 400, 16)	416
maxpool2D_00 (MaxPooling2D)	(None, 150, 100, 16)	0
conv2D_01 (Conv2D)	(None, 150, 100, 24)	9624
maxpool2D_01 (MaxPooling2D)	(None, 37, 25, 24)	0
conv2D_02 (Conv2D)	(None, 37, 25, 32)	19232
maxpool2D_02 (MaxPooling2D)	(None, 9, 6, 32)	0
flatten (Flatten)	(None, 1728)	0
dense_00 (Dense)	(None, 64)	110656
dense_01 (Dense)	(None, 32)	2080
dense_output (Dense)	(None, 20)	660

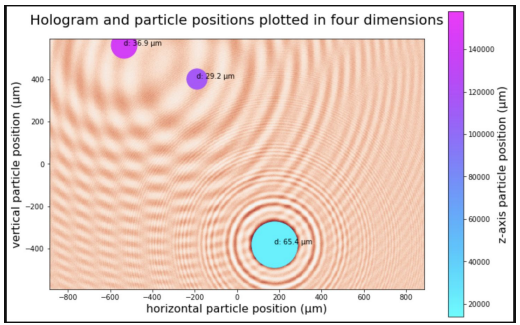
Total params: 142,668
Trainable params: 142,668
Non-trainable params: 0

- Ranked Probability Score
  - Our model: 4.03e-3
  - Uniform distribution: 4.06e-3
  - Ranked Probability Skill Score: 0.007

Team 11: Holodec

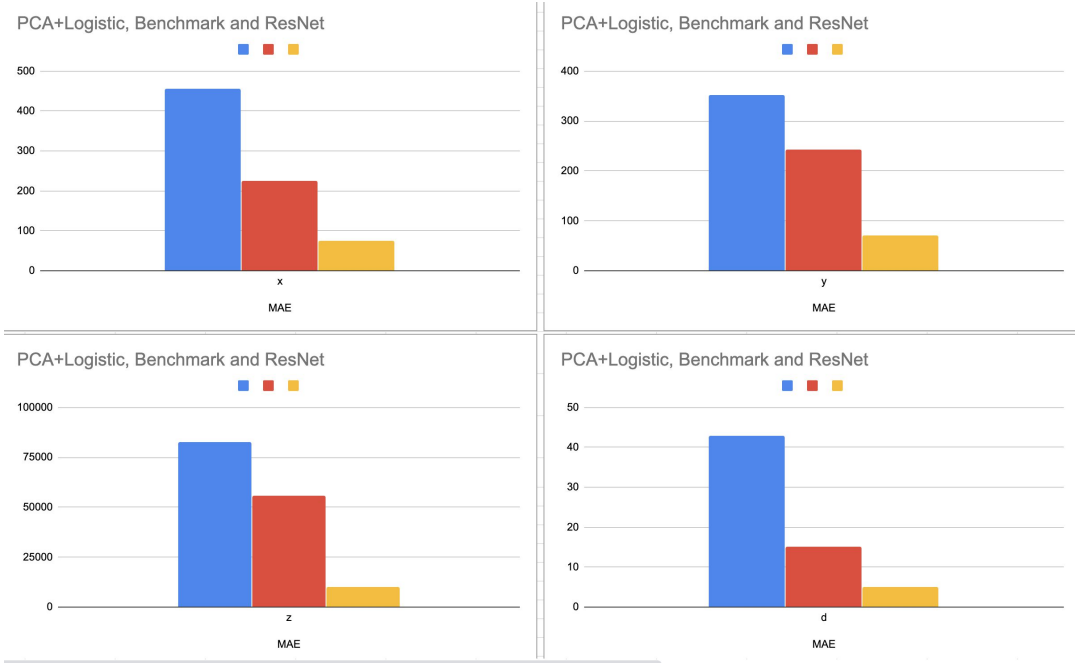
Team Member: Jingyin Tang, Jivesh Dixit, Rosa Paccotacya

- We tried following models on the 1-particle problem
  - PCA + Logistic regression
  - Hough search algorithm
  - Benchmark model, modified benchmarked model and ResNet
  - YOLO (you only look once), but not completed
- A visualization of the data

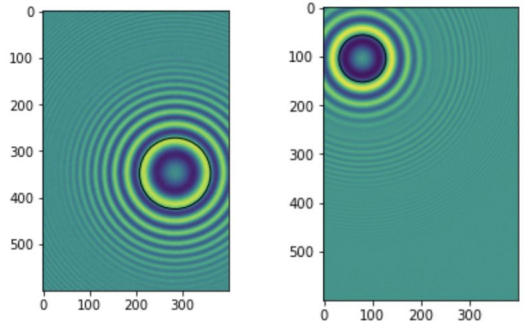


Team 11: Holodec

MAE of PCA+Logistic, Benchmark and ResNet



Notably, with truncated training set (6000 samples), the benchmark model's performance significantly degraded



Challenges:

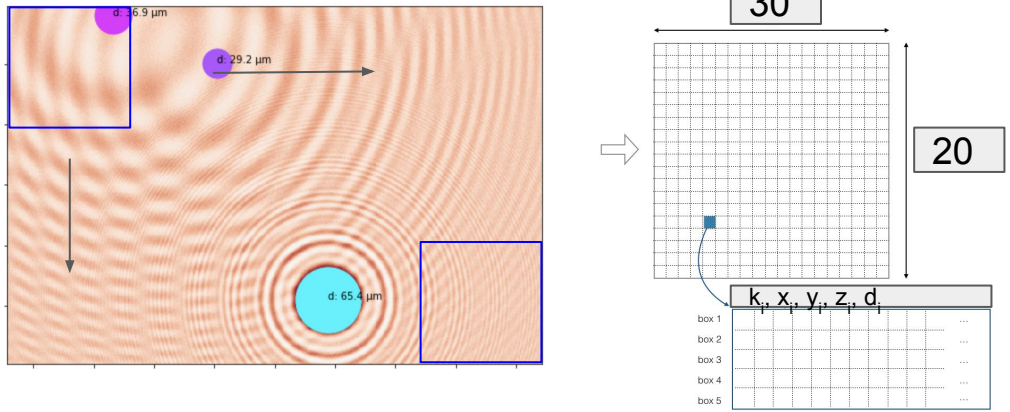
1. Get environment working spends a lot of time. Kernels were often killed during training, caused data loss and progress delay.
2. Hyperparameter is difficult to balance.
3. Preparing input dataset for different kinds of algorithm is more difficult than expected.

Team 11: Holodec

Incompleted trial: YOLO (You only look once)

YOLO perform object identification and segmentation (bounding) in one run.

1. 30x20 slide window; 2, Convolute original image to 30x20xN array. 3 Each 1x1xN box is labeled with (k_i, x_i, y_i, z_i, d_i) where i = 1..num_of_particle; k_i = 1 if have ith particle



Why we cannot complete:

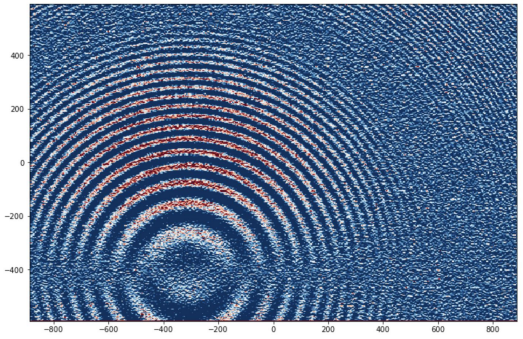
1. Need implement loss function combines IOU and Cross Entropy and non-max suppression
2. Need recreate all training label
3. Jupyterhub died with more than 6000 samples, but not enough.

Team 19: HOLODEC

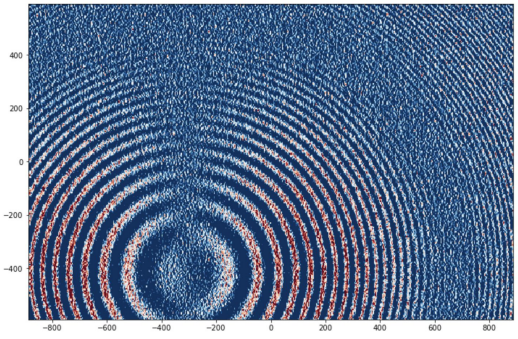
Team Member Names:
Akira Sewnath, Anna Jaruga, Ken Dixon, Sunyoung Kim*

Methods tried:

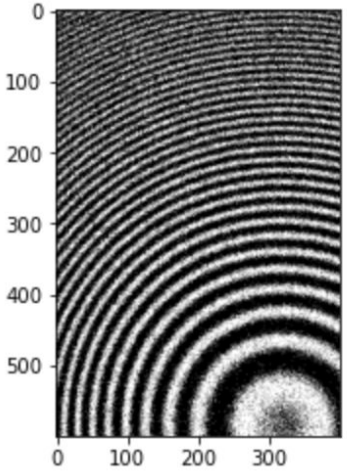
- Linear Regression
- Random Forest
- Gradient Boosting Regressor
- Convolutional Neural Network



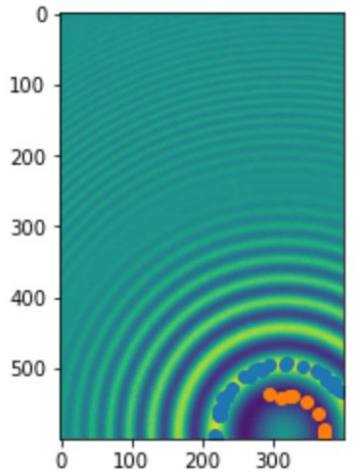
x-gradient of a hologram



y-gradient of a hologram



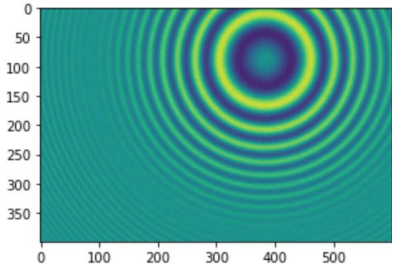
binary filter applied to hologram for feature extraction



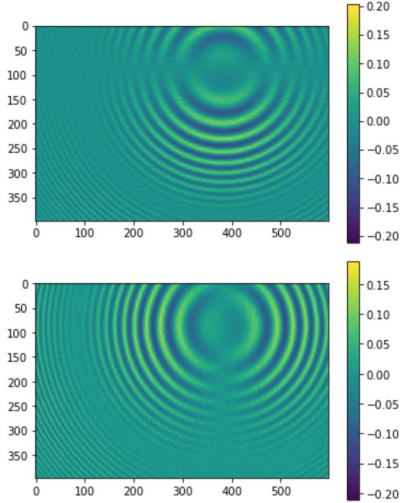
max (orange) and min (blue) distributions in a hologram

Team 19: HOLODEC

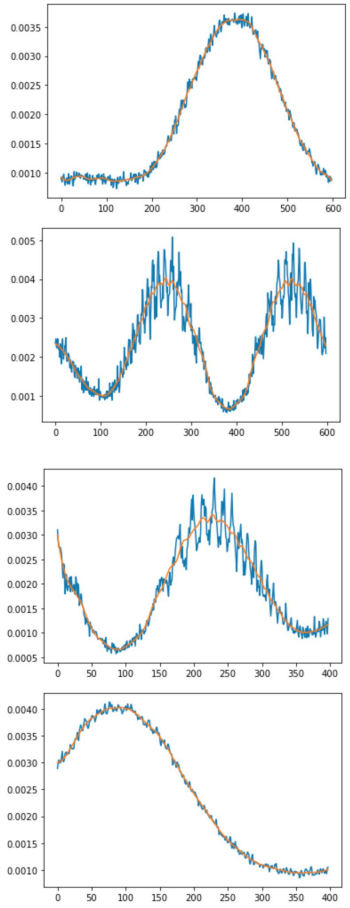
Example feature extraction



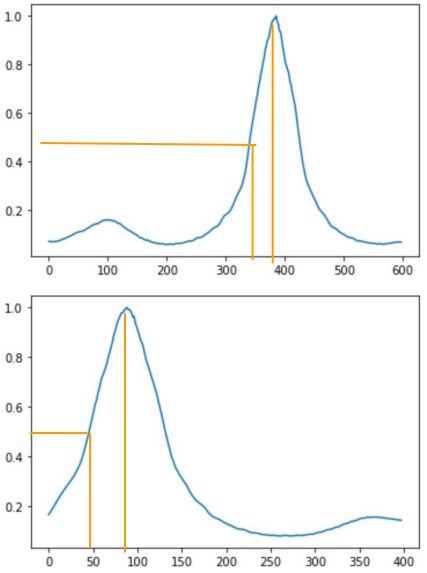
Original image



X and Y gradient



X an Y variance of X and Y gradient



The normalized ratio of x and y variances of gradients

Extracted data:

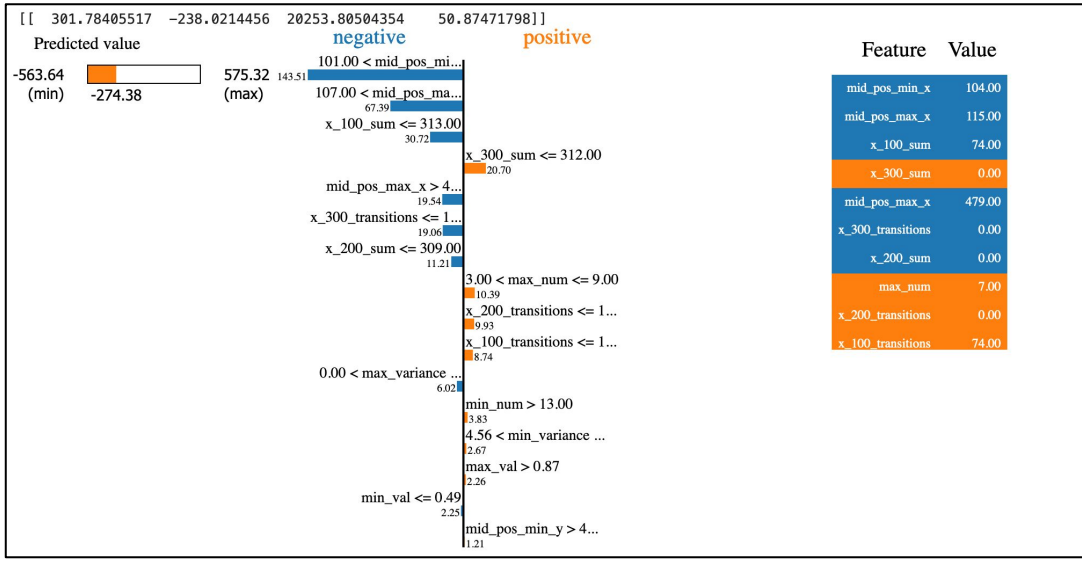
- Location of max ratio
- Distance from max ratio to ratio = 0.5

Team 19: HOLODEC

Model	r^2	MSE
Linear Regression	0.745	5898
Random Forest	0.767	3464
Gradient Boosting Regressor	0.889	3798

Lessons Learned:

- Model interpretability helps find errors in feature extraction
- Intentionally extracting features from the holograms boosted predictions in all the “shallow” models tested
- Sometimes data and computational constraints will not favor deep learning methods

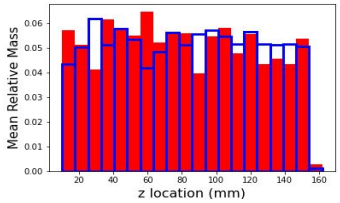


Using LIME to interpret how the Gradient Boosting Regressor predicted y

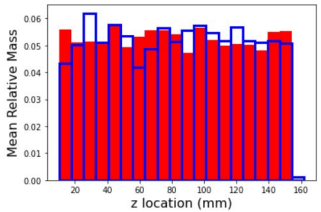
Team 26: Holodec

Team members:
Abigail Whiteside, Burkely Gallo, David Jahn, Sitian Xiong

3-particle model goal: Predict mean relative mass by distance from camera



Baseline
RPSS = -0.234

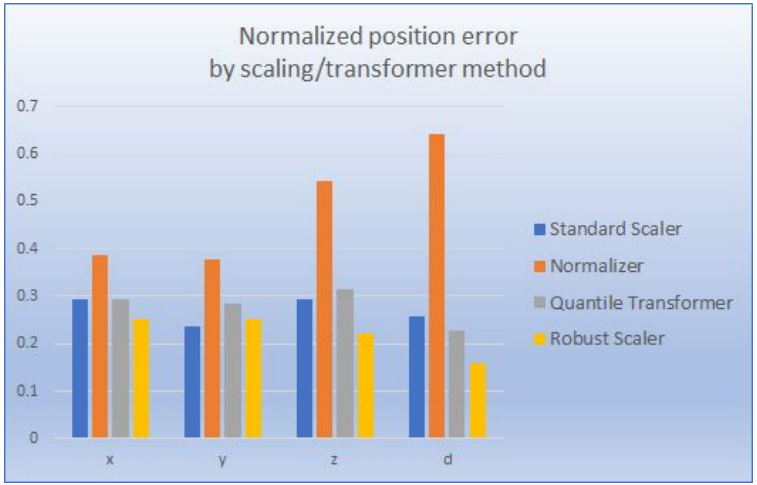


Best fit
RPSS = 0.011

Blue boxes: mean relative mass distribution along z from validation set;
Red boxes: mean relative mass distribution along z from prediction;

Optimize Method	**RPSS**
Baseline Model	**-0.234**
Double dense layer nodes: **[64,32]** to **[128,64]** 	-0.310
Halve dense layer nodes: **[64,32]** to **[32,16]** 	0.009
Batch size increase from **128** to **264** `batch_size = 264`	-0.006
Filter change from **[16,24,32]** to **[4,6,10]** and batch size increase from **128** to **264**	**0.011**
Change filters: **[16, 24, 32]** to **[8, 16, 24, 32]** 	0.000

Team 26: Holodec

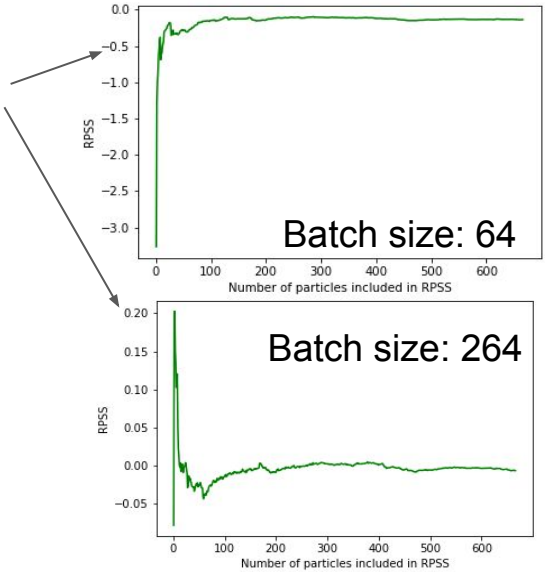


1-particle model goal:
Predict X, Y, Z particle position and diameter (D) of particle (μm)

Robust scaler did best for most metrics, but standard scaler also did very well

Neat Results

- It takes about 200 particles in the three-particle model to asymptote to your final RPSS solution, though this may vary depending on the distribution of particles



Lessons Learned

- Make sure to define the problem and the expected physical results first! This will help with your interpretation of how good your ML model is later.

Challenges

- Find the optimal combination of hyper-parameters for the prediction.
- Connect the hyper-parameters and model to mechanical and physical process.

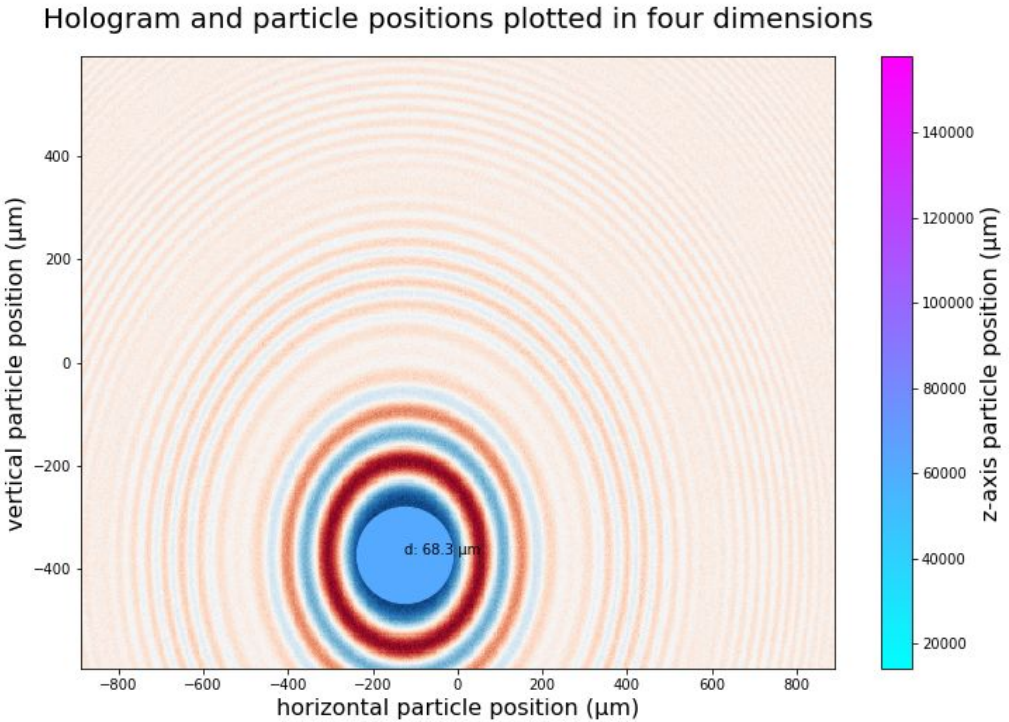
Team 35: Holodec

Don Hood
Vikram Ravi*
Bhanu Magotra*
Angela Bliss
Imoleayo Ezekiel Gbode*

- Data are synthetic images of spherical particles of known size and location
- Baseline method is a convolutional neural network

Attempted techniques

- Kernels of non-square shape (tall and narrow vs wide and flat)
- Altering batch size, training epoch, and activation function



Team 35: Holodec

Did not improve model noticeably

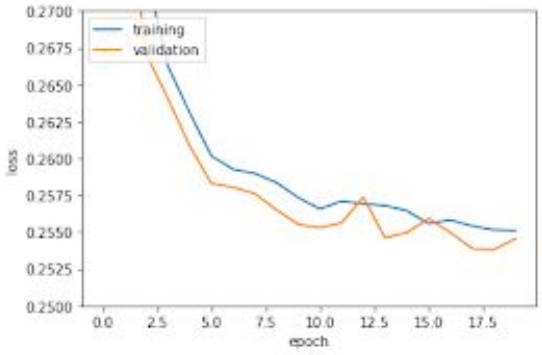
Lessons learned/challenges

- Learning curve of Keras and interpretability was challenging
- Little improvement in error metrics suggest other model modifications are needed

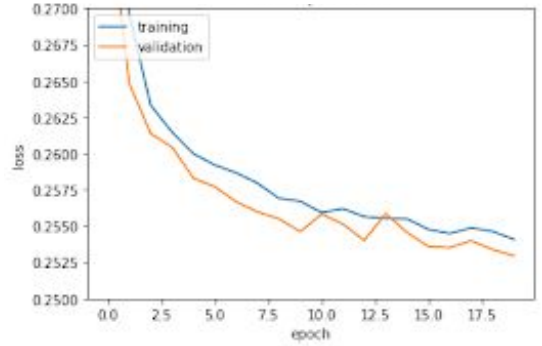
Challenges

- Improving on initial model was challenging
- Model training time was big inhibitor to model experimentation

Baseline Model Loss



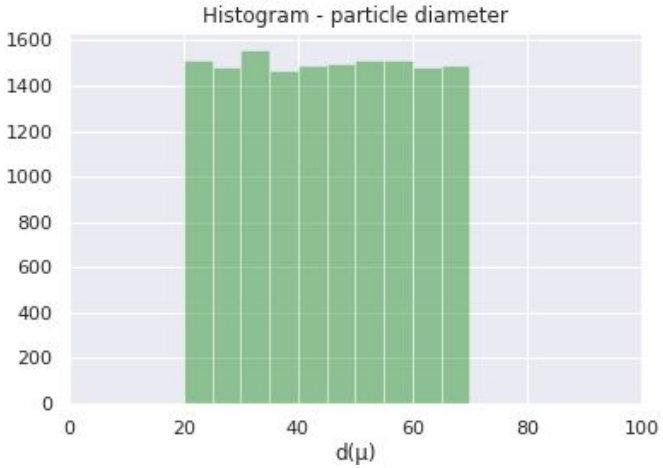
Modified Model Loss

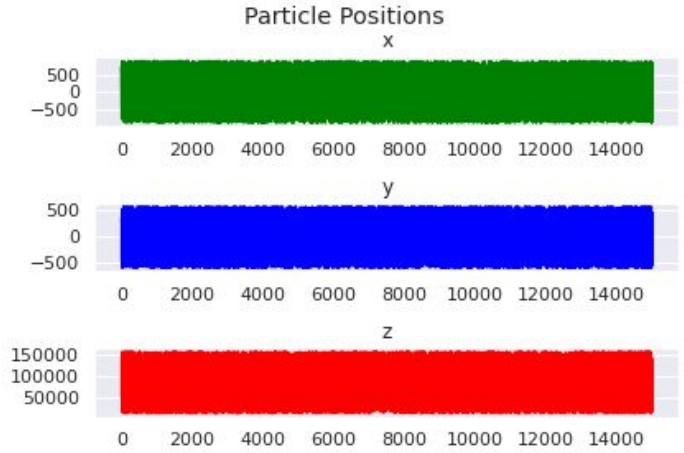


Team 40: HOLODEC (one particle)

Team Members: Alyson Douglas, Paula Marangoni, Qing New

4 Layer CNN

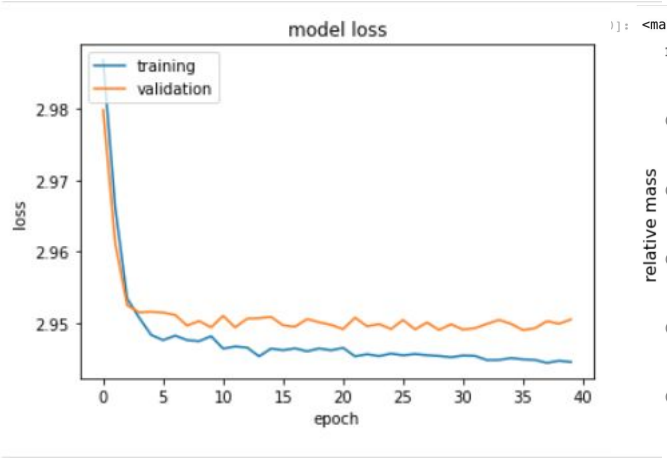


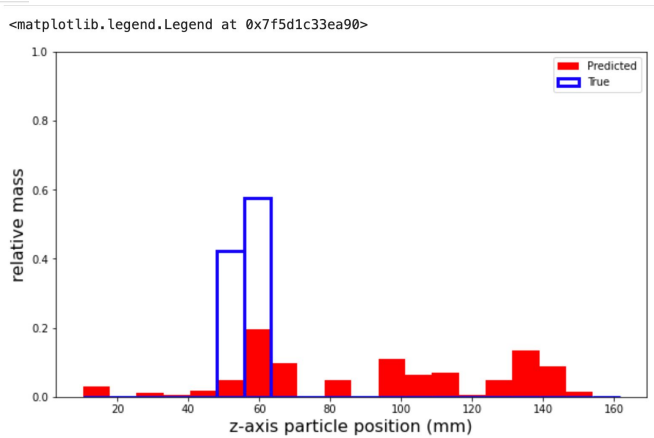


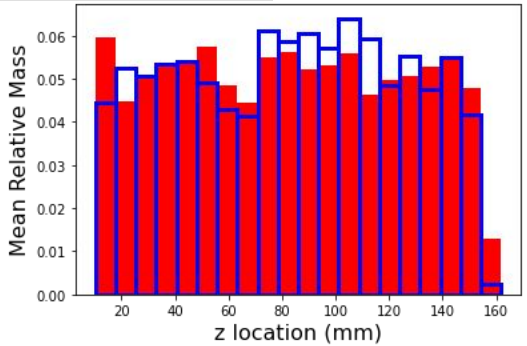
ResNet-50 (could only do 2 epochs with time constraints)

```
x MAE: 622 μm          Max Error: 1,459 μm
y MAE: 297 μm          Max Error: 940 μm
z MAE: 52,044 μm              Max Error: 118,729 μm
d MAE: 18 μm           Max Error: 41 μm
```

Team 40: HOLODEC (three particles)







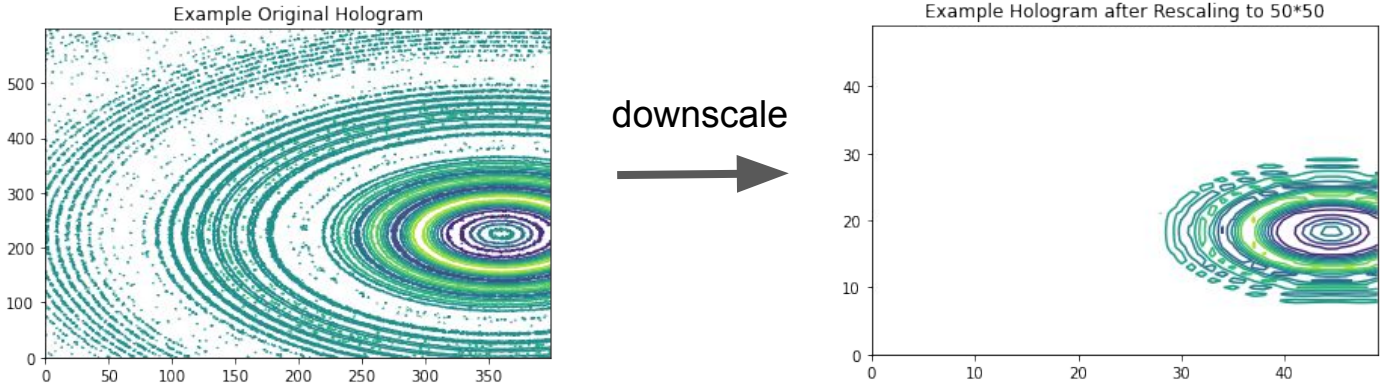
Lessons Learned:

- Hyperparameters make (or break) a model
- More complex models may increase accuracy given **time** and **GPUs**
- Low loss doesn't imply high accuracy
  - Simple CNN had loss < .09, but MAE 100x larger than cut offs
- More particles, more problems

Team 47: <Holodec>

- Team Members:
  - Xia Sun, Sarah Feron, Victor He, Katherine He
- Summary of methods tried
  - Downscale the images from 600*400 to 50*50 to reduce training time:

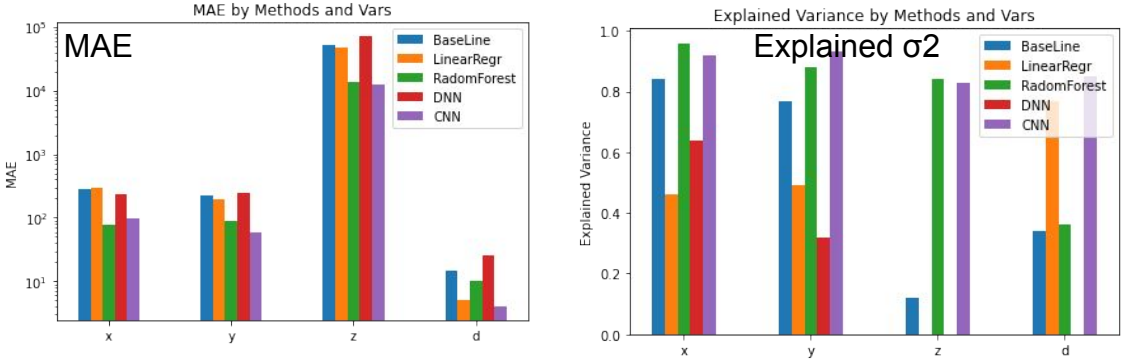
```
from skimage.transform import downscale_local_mean
train_inputs_scaled_one_rescaled=downscale_local_mean(train_inputs_scaled_one,(1,12, 8,1))
```

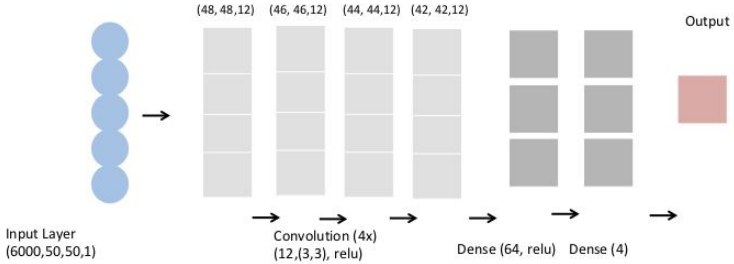


  - Models trained: PCA, LinearRegression, RandomForest, DNN, CNN

Team 47: <Holodec>

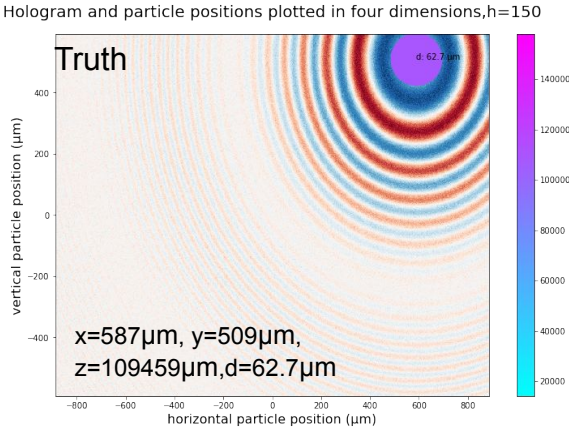
Performance Metrics

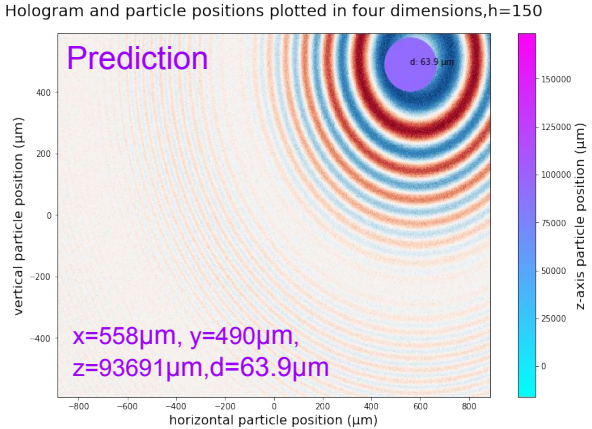




- CNN model (in purple) performed best with lowest MAE and highest explained variance score (avg.0.87), followed by RandomForest (avg.0.76)

Validation example: CNN predicted at h=150





- The validation example using our best model CNN predicted x, y, z, and d (`valid_preds`) agree well with truth (`valid_outputs`)
- Lessons learned/challenges:

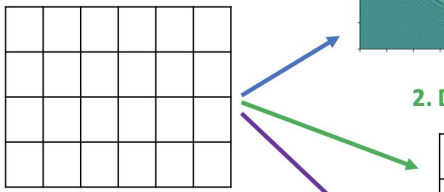
To understand what the individual layers are actually doing and how to come up with a good architecture (rather than trial and error)

Team 49: HOLODEC

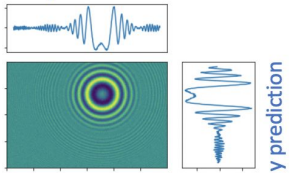
Antara Banerjee, Xinchang Li, Geneva Gray, Narges Shahroudi, and Tenzin Yangkey

Dimensionality Reduction

- We focused on the **single particle problem**
- We attempted first to use simple models and beat the scores from the baseline CNN, and then to tune neural networks
- The challenge in running simple models was the large initial feature space (600x400 pixels)
- We first show 3 methods of dimensionality reduction we used to tackle this problem



1. AVERAGING over x or y dimension

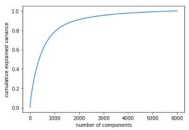


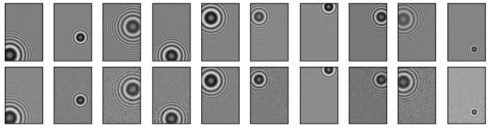
- Worked best for **x and y** prediction
- Mean over y when training for x (reduced features to 600)
- Mean over x when training for y (reduced features to 400)
- This removes **noise** in the direction not being used for prediction
- Worked poorly for diameter prediction (makes sense as we need 2D field for diameter prediction; see below)

2. DOWNSIZING

- Used a factor of 4 to downsize (take 1 pixel for every 4 pixels)
- Worked best for **diameter** prediction
- Worked poorly for x and y prediction (did not get rid of noise as in Method 1!)
- Might improve diameter and z predictions if we center the image using x and y predictions?

3. PCA





- Retained 900 components (>80% cumulative variance)
- Combined with RF: worked best for **diameter** prediction, competitive performance for x & y predictions.

Team 49: HOLODEC

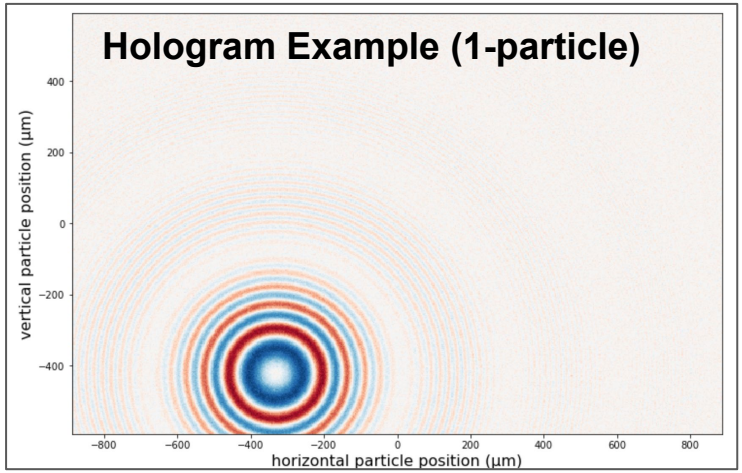
MAE scores for validation predictions

	Baseline CNN 6000 training ~10 mins	Baseline CNN 15000 training NB. CRASHES these numbers are copied from workbook	Simple model, best predictions 6000 training 1 min (ridge), 6 mins (RF)	Simple model, best predictions 15000 training 17 mins
x	~300	20	35 (Mean y, Random forest, default HPs)	23 (Mean y, Random forest, default HPs)
y	~220	12	24 (Mean x, Random forest, default HPs)	16 (Mean x, Random forest, default HPs)
z	~55000	2519	-	-
d	~15	1	4 (PCA, Random forest, default HPs) 5 (Downsize, Ridge regression, α=10)	4 (Downsize, Ridge regression, α=10)

We note that:

- Our predictions from **relatively simple models** trained on 6k images are pretty good compared to a complex CNN trained on 15k images.
- Moreover these simple models are **quicker and less memory intensive.**
- We achieved some improvements on CNN with 6k training size but nowhere near the values for 15k samples – perhaps better to use a different type of NN (e.g. segmentation) for smaller training size?

Team 49: HOLODEC



	1-particle	3-particle
Convolutional NN	Attempt	Attempt
Dense NN	Attempt	No Attempt
Random Forest	Attempt	No Attempt
Linear Regression	Attempt	No Attempt
Gradient Boosting	Attempt	No Attempt

Lessons Learned

- CNN and DNN take a LONG time to run.
- There seem to be unlimited options to reduce dimensionality in the data and tune model hyperparameters. This is not a “one and done” process.
- Understanding the underlying data goes a long way in designing proper ML methods.
- Get familiar with the dataset and try to approach it creatively - may help simplify the problem tremendously