Peak Identification in γ-γ matrices

Jessie Fu ( )
Department of Physics and Astronomy
University of

Yukiya Saito ( )
Department of Physics and Astronomy
University of

Freek De Haas ( )
Department of Zoology
University of

Abstract

The analysis of 2D-histograms plays an important role in experimental nuclear physics. Understanding the location and size of peaks in a γ-γ matrix provides crucial information of the nuclear structure being analyzed. A γ-γ matrix is a 2D-histogram that describes the multiple emission of γ-rays by a nucleus. We implement (robust) Principal Component Analysis, Logistic Regression, and Convolutional Neural Network and compare the models in their effectiveness to identify peaks in a γ-γ matrix. All the peaks in our data were successfully identified independent of the number of peaks or their location, which suggests a solution to the difficulties in the previously proposed method.

1 Introduction

In the field of experimental nuclear physics, and especially in γ-ray spectroscopy, a large fraction of time is consumed by analyzing 2D-histograms. Identifying the peaks and their characteristics plays an important role in analyzing these histograms. The location of these peaks provides useful information about the radioactive decay as well as the structure of the nucleus being analyzed. Such 2D-histograms are constructed by recording the number of counts of the energy of one γ-ray relative to the energy of the reference γ-ray within a certain time window (typically $\sim$2 μs). These 2D-histograms are usually called "γ-γ matrices". The detection system consists of a multi-channel high-purity germanium (HPGe) detector array, such as the GRIFFIN spectrometer, located at TRIUMF on the UBC campus (Figure 1). A multi-channel detection system allows the detection of multiple γ-ray emission at the same time, *i.e.* γ-rays in coincidence. By combining the information obtained from the γ-γ matrix with the theoretical description of the nucleus, one can reconstruct the level scheme (*e.g.* Figure 2), which describes the level structure of a nucleus.

[Figure 1 about here.]

For example, the radioactive decay of ^{60}Co (the decay scheme is shown in Figure 2) is one of the most commonly used for γ-ray calibration. It generates a 2D-histogram shown in Figure 3.

[Figure 2 about here.]

[Figure 3 about here.]

One of the problems regarding the current analysis methods of 2D-histograms (γ-γ matrices) is that the features of 2D-histograms are not easy to recognize. Often they are sliced and projected onto one axis to generate 1D-histograms, which are much more human-readable. However, slicing 2D-histograms means discarding most of the global information on the 2D-histograms and it is possible that such a dimension reduction leads to a biased and inaccurate analysis.

2 Related work

Previously, there have been three papers [Morháč et al., 1997a,b, 2000] discussing the direct analysis of γ-γ matrices and those are the only ones which are implemented in the commonly used analysis framework ROOT [Brun and Rademakers, 1997]. This implementation provides background reduction, peak deconvolution, and most importantly peak detection functionality. However, running this program is computationally very expensive and it has to be run for each γ-γ matrix. This program is unsupervised, using non-adaptive algorithms. The hyper parameters for each algorithm have to be manually optimized by running the program multiple times. Moreover, the peak detection algorithm can store only up to 10 peaks, which is not enough since experimental data can involve hundreds of peaks on a γ-γ matrix. Another issue regarding the peak finding algorithm is that it detects the crossing points of diagonal lines and vertical (horizontal) lines as peaks (see Figure 4 for example). Due to these difficulties, this program is not widely used for actual research.

[Figure 4 about here.]

3 Project description

The algorithm discussed in Section 2 was proposed in the late 90's. Considering the rapid advancement of machine learning research, there might be algorithms which can process γ-γ matrices more efficiently and effectively. More specifically, the objective of the current project is to develop a method which can directly extract information from γ-γ matrices to identify and locate peaks independent of the number of peaks or background structure. Primarily, the focus is to detect the location of peaks in the cropped 2D-histograms (e.g. Figure 5a) while being insensitive to background structure. With background structure we refer to the vertical, horizontal, and diagonal lines as shown in Figure 2.

3.1 Data

The data set used in this project is a set of cropped γ-γ matrices. The cropped regions are 128×128 pixels and they were hand picked so that each example contains one or more peaks in order to encourage the training of the algorithms. The original γ-γ matrices have dimensions of 6000×6000 and the size of the peaks are very small relative to the dimensions of the original matrix. Some of the set of cropped matrices have partial overlap, which means that not all examples are completely independent.

Figure 3 shows a typical example of a γ-γ matrix with three main components: peaks, lines (vertical, horizontal, and diagonal), and background. It is not impossible to identify the peaks just by looking at the histograms. However, it is very difficult to distinguish the real peaks from the crossing of two lines. Another reason for the difficulty in identifying peaks is that, depending on the regions of the histograms, the height of background is very different due to the physical process called Compton scattering, which is more prominent in the region smaller than 1000 keV (kilo electron volt). Moreover, the height of the peaks also varies depending on the intensity of the γ-rays. However, the three basic components discussed above are common for any kind of γ-γ matrix. Machine learning techniques are generally good at capturing the common patterns of the input data. Therefore, if appropriate machine learning algorithms are applied, they may pick up such common features. Furthermore, it is expected that they are able to identify peaks in a way that it is insensitive to the location or the strength of the background.

The training examples were generated from an actual measurement using the ^{56}Co source with the aforementioned GRIFFIN spectrometer. The label (ground truth) corresponding to the actual data (Figure 5a) was produced manually by setting the thresholds for the number of counts in the 2D-histogram (Figure 5b).

[Figure 5 about here.]

3.2 Proposed Methods

The proposed methods are the following: Principal Component Analysis (PCA) as well as robust PCA, Logistic Regression, and Convolutional Neural Network (CNN).

We expect the background, lines and the presence of peaks to be common in all γ-γ matrices, and the location and intensity of the peaks to be unique. PCA might be able to extract common background patterns in the first few principal components (PC). Residuals of PCA may contain information about the location and intensity of the peaks. Because the peak signal is very sparse, robust PCA might allow us to compute a sparse representation of the location of the peaks. If we assume that the background is common in all 2D histograms, we can treat the peak as the foreground that is different in each 2D histogram.

To do logistic regression, we learn and predict on each individual pixel using its neighbours as features (Figure 6). We reason that the direct surroundings of a pixel is a key determining factor to differentiate noise from signal.

As for CNN, since each data point in the γ-γ matrices are strongly correlated with neighbouring pixels, CNN would be a reasonable choice to handle this type of data. There have been already various application of CNN to identify the location of specific objects in a given image. In the case of the current project, the goal is to detect peaks on a γ-γ matrix as well as identify their location.

3.3 Logistic Regression

3.3.1 Feature extraction

For this part of the analysis we used $n = 20$ gamma-ray spectroscopy figures of dimensions 127×127. First we split the data evenly in a labeled training- (X_{train}) and test (X_{test}) set where each element in the set corresponds to a γ-γ matrix. We restrict our attention to the pixels that have neighbours and ignore borders. We create a transformed training set where each pixel in the original figure (except the borders) is represented by a row in the transformed training set. The number of features in the transformed training set will depend on the number of neighbours you include in the features: $n_{train} \times (127 - n_{neighbours})^2$. In order to validate the model we apply the same transformation to the test set.

[Figure 6 about here.]

3.3.2 The model

For training we take a random sub-sample of the training data ($n_{sample} = 10000$) because the total training data is too large to train on. We trained a logistic regression model in Python using "scikitlearn" tools [Pedregosa et al., 2011]. We used the "LBFGS" solver with L2-regularization and adjusted the weights inversely proportional to class frequencies. This was necessary because class 0 and 1 are not at equal frequencies present in the sample. It drastically improved the performance of the model. We fixed the regularization parameter to 1. In the future we would like to treat it as a hyperparameter, however there was not enough time to implement this.

Model training and prediction is slow, because we need to extract the features and effectively predict about 127×127 times to recreate an original gamma-ray figure.

3.4 Principal Component Analysis

We applied PCA with Singular Value Decomposition algorithm (SVD) to the 20 training examples using Matlab build-in function (pca). More specifically, we first reshaped each 127×127 2D histogram into a long row vector ($127 \times 127 = 16129$ elements). 20 row vectors were then concatenated to form a $n \times p$ matrix ($n = 20$ and $p = 16129$). PCA was applied to the centered, log-transformed $n \times p$ data matrix X. After choosing the number of PCs to keep based on the percentage variance accounted for by each PC, we reconstructed the data with the kept PCs ($\hat{X} = ZW$, where Z is principal component scores and W is the principal components). The residuals matrix R was calculated as $X - \hat{X}$.

3.5 Robust Principal Component Analysis

We applied robust PCA using open source Matlab code (*https://github.com/dlaptev/RobustPCA*) to decompose the input matrix X into common background (L) and foreground (S), where L is low rank and S is sparse. The locations of the non-zero elements in S might give the location of the peaks. The implemented method uses Principal Component Pursuit (PCP) estimate [Bertsekas, 2014, Candès et al., 2011]:

$$\text{Minimize: } ||L||_* + \lambda ||S||_1 \tag{1}$$

$$\text{Subject to: } L + S = X \tag{2}$$

where $||.||_*$ is a nuclear norm, $||.||_1$ is L1-norm.

This PCP problem is solved using an iterative augmented Lagrange multiplier (ALM) algorithm [Yuan and Yang, 2009]:

$$l(L, S, Y) = ||L||_* + \lambda ||S||_1 + < Y, X - L - S > + \frac{\mu}{2} ||X - L - S||_F^2 \tag{3}$$

Details of this algorithm are beyond the scope of this project. To summarize, the algorithm first minimizes l with respect to L (fixing S), then minimize l with respect to S (fixing L), then update Y (Lagrange multiplier matrix) based on the residual $X - L - S$. Hyperparameters were set to: regularization parameter $\lambda = 1/N$, augmented lagrangian parameter $\mu = 10\lambda/3$.

To examine the prediction accuracy, we first normalized each row of S by its maximum. To visualize the the location of the predicted peaks, we thresholded S by 0.25 to create binary predicted labels ($\hat{y}$=S_{binary}). We then computed the centroid and radii of clusters with at least 8 connected components in each $\hat{y}$. We then overlaid circles with the same centroid and radii onto the true label maps y.

For quantitative measurement, we formed the confusion matrix from S_{binary} and y for different thresholds. We then calculated the accuracy, precision, true positive rate and true negative rates at each threshold.

3.6 Convolutional Neural Network

The network architecture we adopted in the current project is illustrated in Figure 7. This was proposed by Ronneberger et al. [2015] and is called U-net. We used one of the publicly available implementations of U-net, which is available at `https://github.com/zhixuhao/unet`. The original purpose of the network was biological image segmentation, but it is also used in other fields such as astronomy. The basic structure is the layers of 3×3 convolutions followed by a rectified linear unit (ReLU) activation and a 2×2 max pooling. After this contracting path, upsampling is done by a 2×2 convolution. In addition to this, a cropped concatenation from the downsampling part is applied in order to restore the loss of border pixels. At the final 1×1 convolutional layer, each feature vector is mapped to the labels.

[Figure 7 about here.]

20 of labeled images with the size of 526×526 pixels were trained with the stochastic gradient descent using adaptive moment estimation method (Adam) by the minimizing binary cross-entropy:

$$E(\mathbf{w}) = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)], \tag{4}$$

where $\mathbf{w}$ is a weight vector, N is the number of examples, y_i is the label, and $\hat{y}_i$ is the predicted label. The optimization was iterated 300 times during one epoch. In addition to the original 20 labeled images, each image was augmented by rotation and translation in order to encourage invariance and robustness properties. For this experiment, a MacBook Pro with a 2.7 GHz Intel Core i5 CPU was used. The training took 2547 seconds and the prediction took 2 s per image.

4 Results

4.1 Logistic Regression

We explored different feature extraction methods with varying success. We tried a range of $n_{neighbours}$ from 0 to 12. We see that when we increase $n_{neighbours}$ the model becomes more accurate in recognizing peaks (Figure 8).

[Figure 8 about here.]

Figure 9 shows the predictions on two test sets where our model was most successful. We achieved the most success with $n_{neighbours} = 10$. We see that in (b), where the two bands cross, the model does not predict a signal, even though the absolute value is higher than for the bottom signal in (a). This suggests that the model uses the neighbourhood pattern rather than the absolute values.

Furthermore, we tested the different models with different $n_{neighbours}$ on 10 test examples and calculated their test-score in the following table:

$n_{neighbours}$	score
0	0.9969
1	0.9970
2	0.9968
3	0.9961
4	0.9869
5	0.9928
6	0.9970
7	0.9928
8	0.9953
9	0.9998
10	0.9912
11	0.9982
12	0.9972

.

[Figure 9 about here.]

Interestingly, if we project the values of the learned coefficients to their appropriate place in the neighbourhood space we see a pattern. Figure 10 shows the coefficients as they represent the neighbours relative to the focal cell (see Figure 6 for example of $n_{neighbours} = 1$). The weights of the coefficients form a spatial pattern with the focal pixel at the center. This pattern is especially obvious in Figure 10-d where $n_{neighbours} = 10$, which was our best performing model (see Figure 8).

[Figure 10 about here.]

4.2 Principal Component Analysis

The first PC accounted for 85.8% of the total variance as shown in Figure11a. The first four PCs (W) reshaped back to the original dimension are shown in Figure12. Scree test showed only the first PC should be retained based on Figure11a and maybe two to six PCs based on Figure11b. Residual matrices highly resembled the input matrices as shown in Figure13. PCA with SVD algorithm removes some common background, but cannot extract the peak for each 2D histogram effectively.

[Figure 11 about here.]

[Figure 12 about here.]

[Figure 13 about here.]

4.3 Robust Principal Component Analysis

The first PC accounted for 90.6% of the total variance. Robust PCA was able to successfully separate the peak in the foreground (S) matrix as shown in Figure14. To examine the quality of the prediction, Figure15 shows the comparison between true labels Y, S (foreground, predicted labels) and clusters defined on S overlaid on Y at threshold 0.4.

[Figure 14 about here.]

[Figure 15 about here.]

Accuracy, precision, true positive and true negative rates were plotted as a function of threshold as shown in Figure16. Mean accuracy and true negative rates for the 20 training samples were greater than 99% at all thresholds (0-0.7). There was a trade off between precisions and true positive rates as we increased the thresholds. This might be due to the fact that our ground truth labels obtained with manual thresholding still captured some false positive signals as shown in Figure15 Example 1 (top left). Clustering methods shown in Figure15 should be able to capture the truth peak locations.

[Figure 16 about here.]

4.4 Convolution Neural Network

We used 11 images as test data which are different from the training data set. An example of the quality of the prediction is shown in Figure 17. The prediction accuracy was calculated by comparing the label (ground truth) and the prediction pixel by pixel, using `sklearn.metrics.imbalanced_accuracy_score()` function. The average accuracy over 11 test example was calculated to be 84.9%, with the standard deviation of 9.2%. Qualitatively, U-net successfully detected all the peaks in the test examples.

[Figure 17 about here.]

5 Discussion and future work

In the current project, we have developed three peak detection methods in γ-γ matrices, which are one of the most common types of data in experimental nuclear physics. In a qualitative manner, all the methods were successful in detecting the peaks. The insensitivity to the crossing points of diagonal lines and vertical (horizontal) lines were also achieved, which shows superiority to the previous methods proposed by Morháč et al. [2000].

As future work, developing peak finding methods which can handle the full size γ-γ matrices is necessary for practical use. Unfortunately we did not have a lot of training data available; our models would likely perform better and with higher accuracy if we had more data. However the time we needed to invest in that would have been outside the scope of this project.

Combining multiple methods proposed in this project may increase the accuracy and possibly lead to more accurate predictions. Specifically, we think that PCA can be beneficial to increase the variance in noise to signal ratio and to apply other methods such as logistic regression or U-NET on the PCA transformed data. Perhaps it would be possible to use logistic regression to identify a convolution filter for CNN as we have seen in Figure 10

The logistic regression is a slow predictor because it requires heavy pre-processing (feature extraction). The U-net implementation however predicted on average within 2 seconds. Depending on the application, one might be preferred over the other.

One issue with our setup is that we break the "Golden rule" in the logistic regression analysis. Because our examples are cropped images of the same $\gamma - \gamma$ matrix, there is partial overlap between some training and test examples. We were aware of this issue but had no time or resources to acquire more data. We therefore make a note of caution with these results.

We are also interested about the patterns in Figure 10. When we project the weights of the logistic regression onto its spatial domain like in Figure 10, they form a pattern where the weight in the center is positive and surrounding it is negative. We hypothesize that it is basically doing the same as a Laplacian filter which are used as a convolution layer in CNNs. It would be interesting to follow up on these results to see if we can use logistic regression to learn convolution filters for CNN.

We believe this preliminary research is a step in the right direction to improve current models to identify peaks in γ-γ matrices.

References

D. P. Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

R. Brun and F. Rademakers. Root — an object oriented data analysis framework. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 389(1):81 – 86, 1997. ISSN 0168-9002. doi: https://doi.org/10.1016/S0168-9002(97)00048-X. URL `http://www.sciencedirect.com/science/article/pii/S016890029700048X`. New Computing Techniques in Physics Research V.

E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

M. Morháč, J. Kliman, V. Matoušek, M. Veselský, and I. Turzo. Efficient one- and two-dimensional gold deconvolution and its application to γ-ray spectra decomposition. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 401(2):385 – 408, 1997a. ISSN 0168-9002. doi: https://doi.org/10.1016/S0168-9002(97)01058-9. URL `http://www.sciencedirect.com/science/article/pii/S0168900297010589`.

M. Morháč, J. Kliman, V. Matoušek, M. Veselský, and I. Turzo. Background elimination methods for multidimensional coincidence γ-ray spectra. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 401(1): 113 – 132, 1997b. ISSN 0168-9002. doi: https://doi.org/10.1016/S0168-9002(97)01023-1. URL `http://www.sciencedirect.com/science/article/pii/S0168900297010231`.

M. Morháč, J. Kliman, V. Matoušek, M. Veselský, and I. Turzo. Identification of peaks in multidimensional coincidence γ-ray spectra. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 443(1): 108 – 125, 2000. ISSN 0168-9002. doi: https://doi.org/10.1016/S0168-9002(99)01005-0. URL `http://www.sciencedirect.com/science/article/pii/S0168900299010050`.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. URL `http://arxiv.org/abs/1505.04597`.

X. Yuan and J. Yang. Sparse and low-rank matrix decomposition via alternating direction methods. *preprint*, 12:2, 2009.

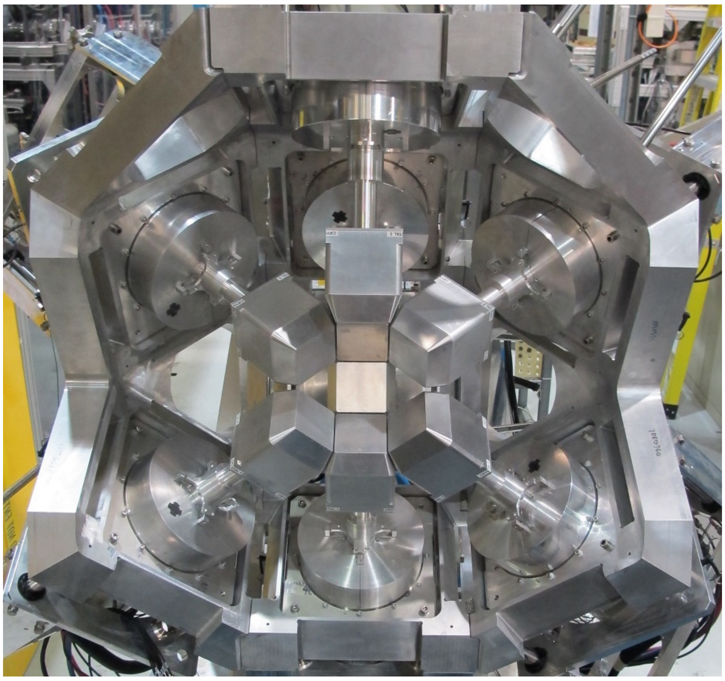


Figure 1: Picture of the GRIFFIN spectrometer located at the ISAC facility at TRIUMF. The γ-ray detectors consist of 64 independent channels, which allows the detection of multiple coincidental γ-ray emission.

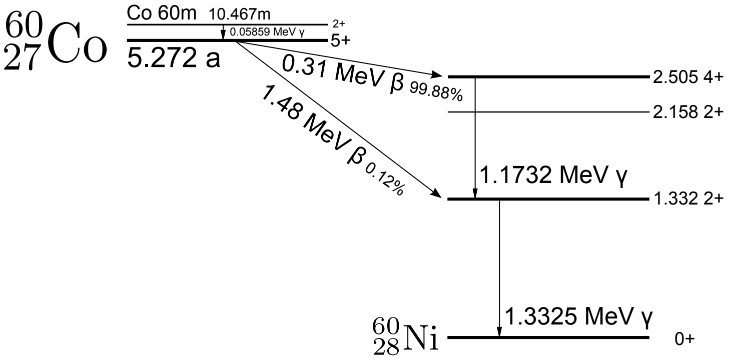


Figure 2: Decay scheme of ^{60}Co. With the probability of more than 99%, ^{60}Co decays into the 4^+ excited states in ^{60}Ni and promptly decays to the 0^+ ground state by emitting two γ-rays with the energies of 1.1732 MeV and 1.3325 MeV (Mega electron Volt). These two γ-rays are in coincidence and produce peaks in a γ-γ matrix. The figure was taken from Wikipedia.

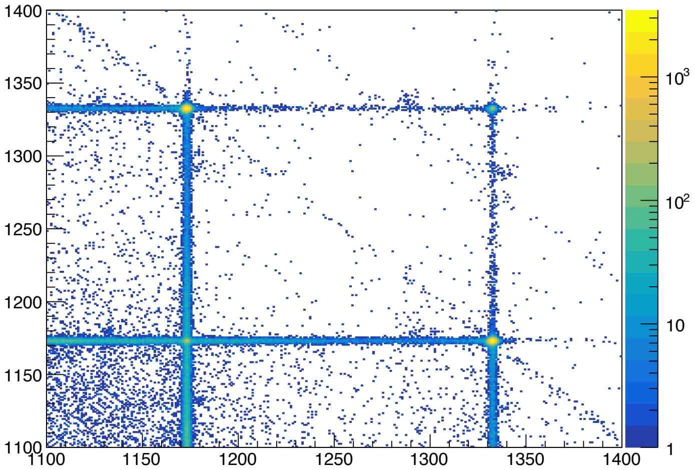


Figure 3: γ-γ matrix of a ^{60}Co calibration source measurement. There is a peak at the intersection of 1173.2 and 1332.5 keV (1332.5 and 1173.2 keV) since those two γ-rays are in coincidence.

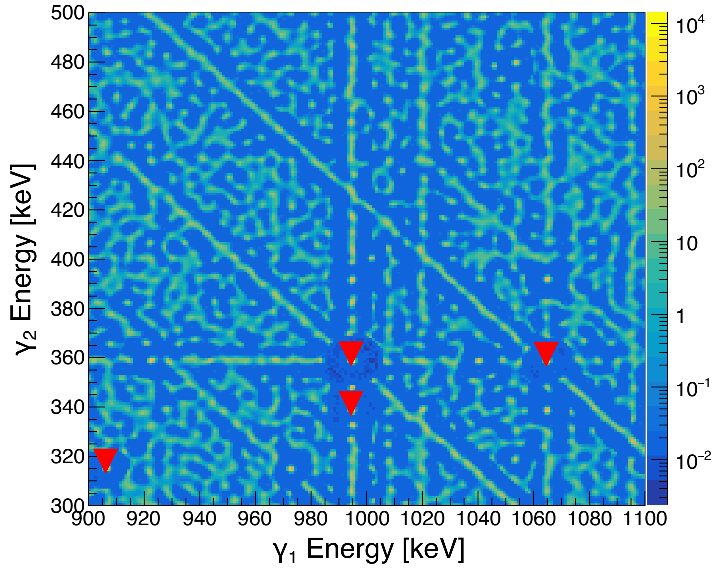
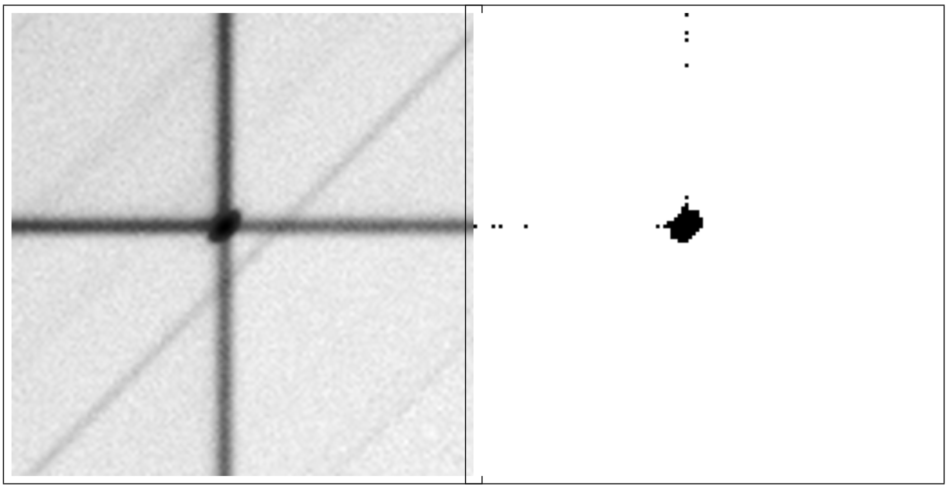


Figure 4: An example of peak search by previously proposed method Morháč et al. [1997a,b, 2000]. The right most triangle is a falsely identified peak.



(a) (b)

Figure 5: (a) An example of a cropped 2D-histogram used in this project. (b) An example of a label for a cropped 2D-histogram corresponding to Figure 5a. This is produced by manually setting a threshold for the number of counts

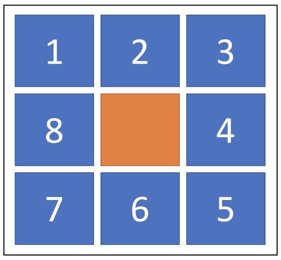


Figure 6: An example of the features (blue) of a focal pixel (orange) when $n_{neighbours} = 1$

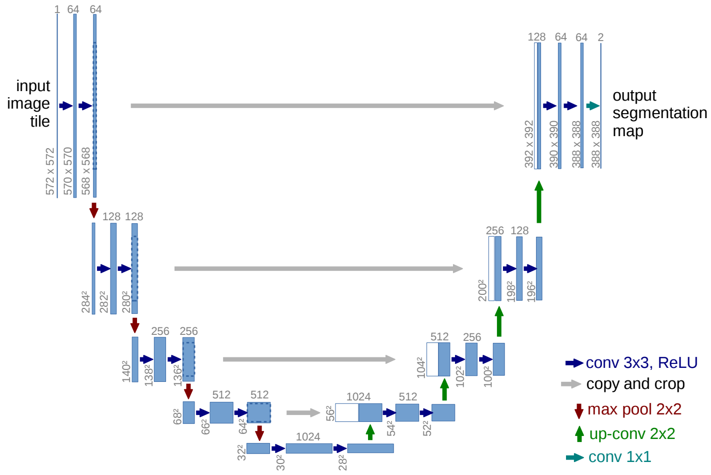
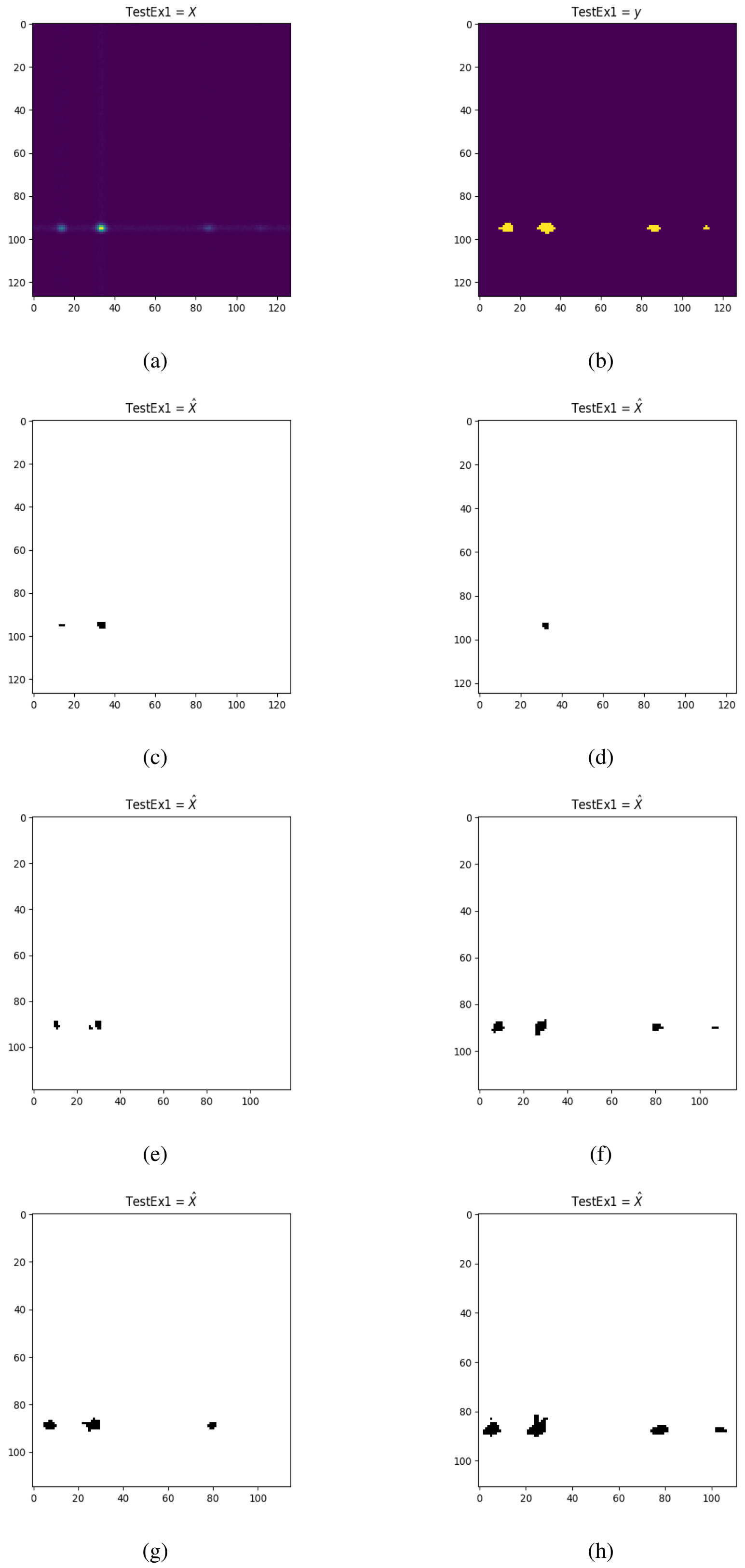


Figure 7: Illustration of the network architecture of U-net. See text for details.



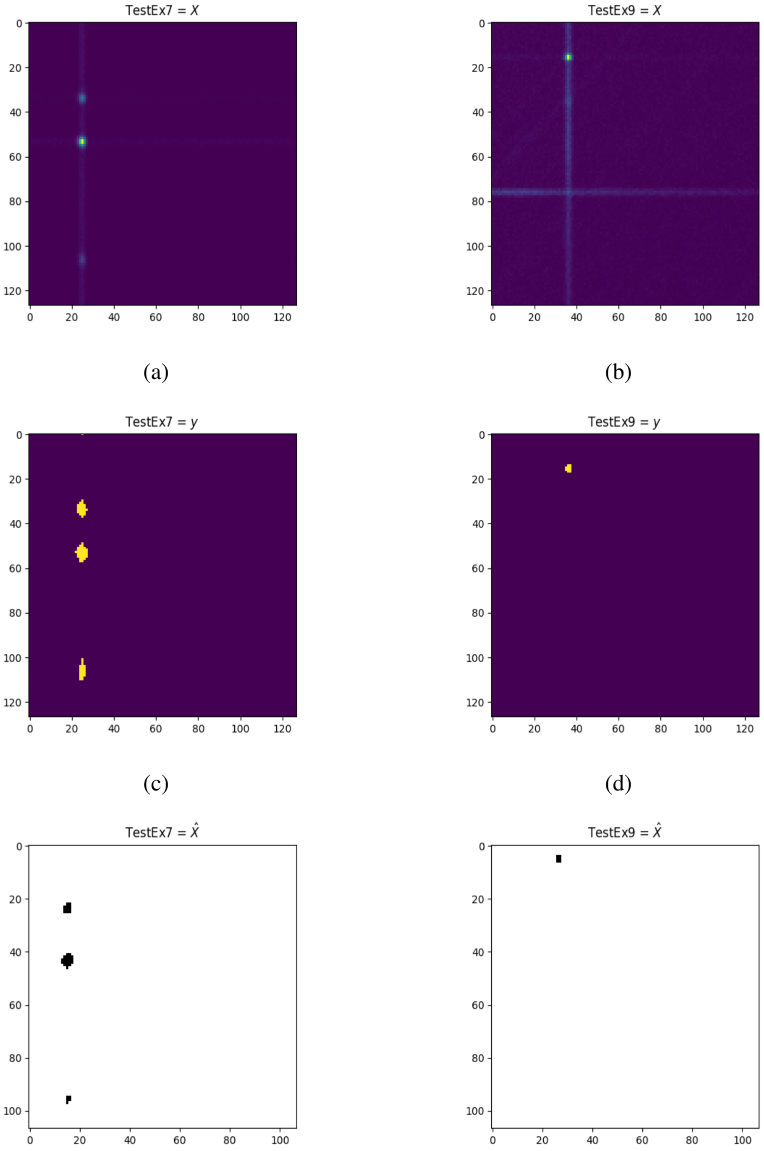
(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 8: a: This is an example of the test data set. b: this is the corresponding labels to (a). c,d,e,f,g,h: Are the predictions of logistic regression by increasing the $n_{neighbours}$ = (c=0), (d=1) , (e=4) , (f=5)) , (g=6), (h=8)



(a)

(b)

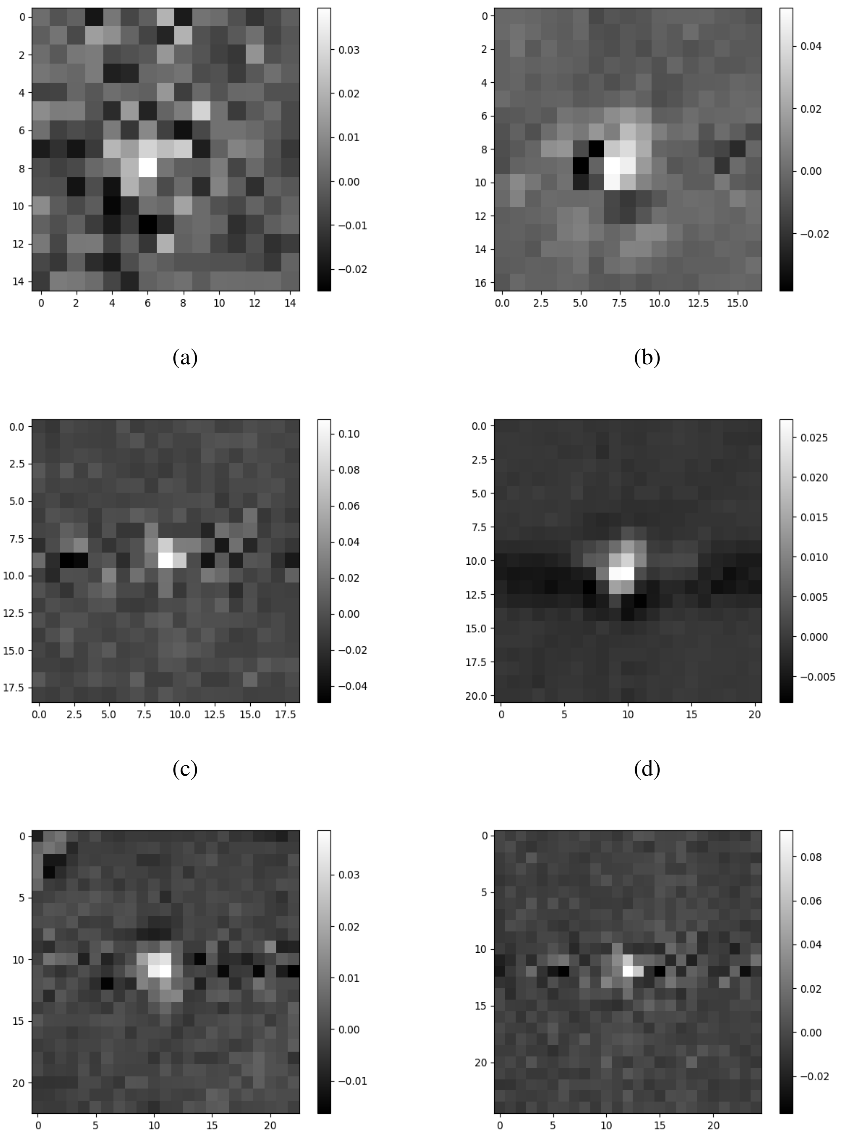
(c)

(d)

(e)

(f)

Figure 9: These are two test examples (the left and the right column) for the Logistic Regression model with $n_{neighbours} = 10$. Top: Input data (X), Middle: Ground truth(y), Bottom: Prediction by logistic regression. The prediction figures are slightly cropped due to the lack of borders



(a)

(b)

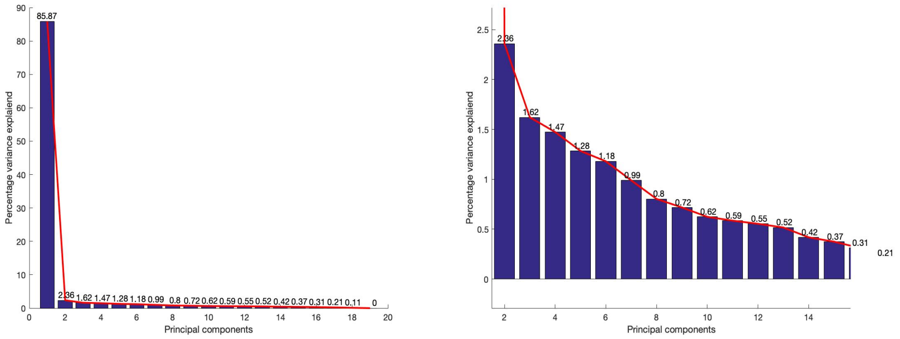
(c)

(d)

(e)

(f)

Figure 10: These are plots of the weight coefficient with at the center the focal pixel. $n_{neighbours} = 7$ (a) ,8 (b) ,9 (c) ,10 (d) ,11 (e),12 (f)



(a)

(b)

Figure 11: Scree plot showing the percentage variance accounted for all principal components (a), and zoomed for principal component 2 to 19 (b)

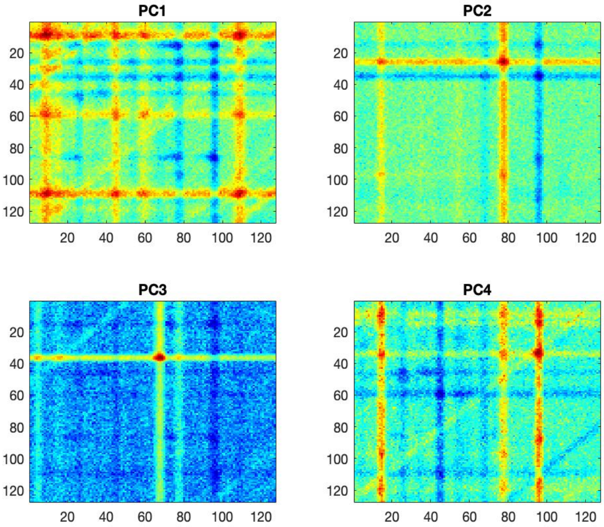


Figure 12: First four principal components reshaped back to the original dimensions.

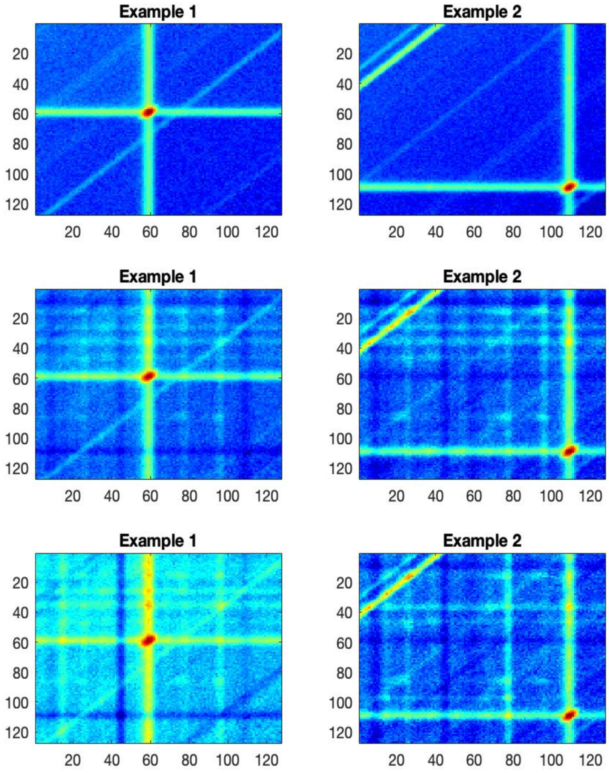


Figure 13: Comparison between input data X (top), PCA residuals using first PC (middle), and PCA residuals using the first six PCs (bottom) .

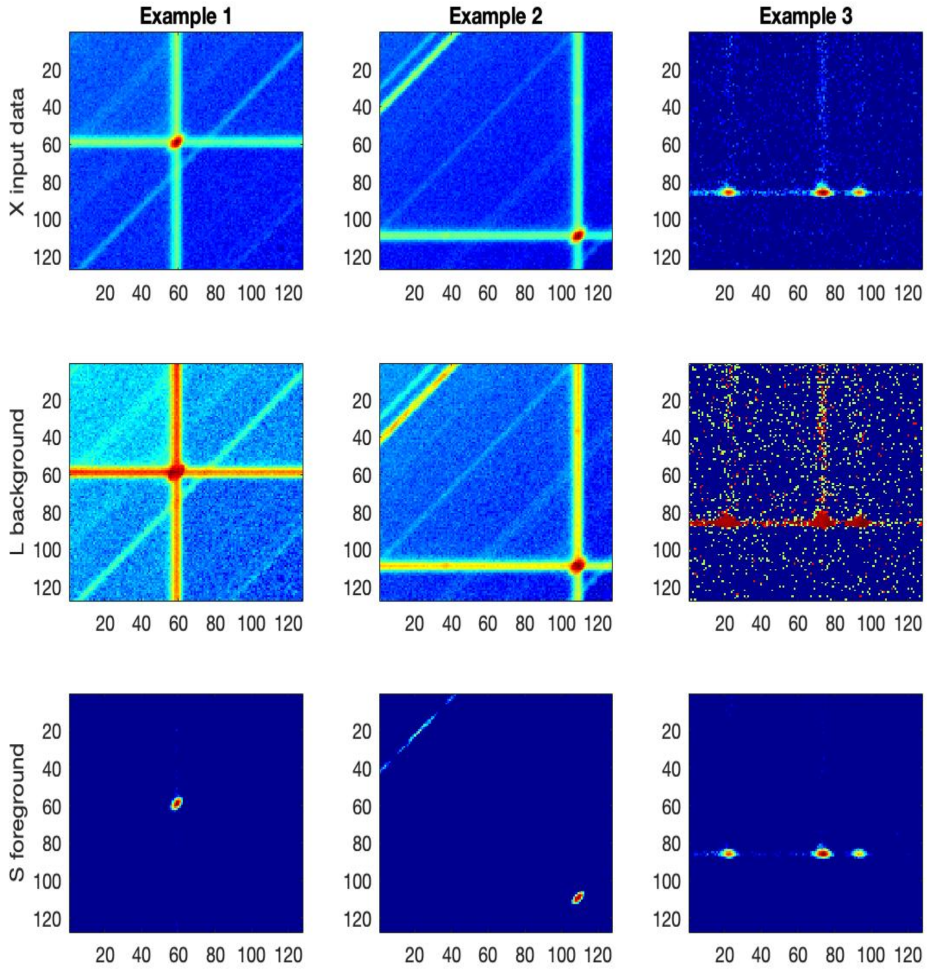


Figure 14: Comparison between input data X (input), L (background), and S (foreground).

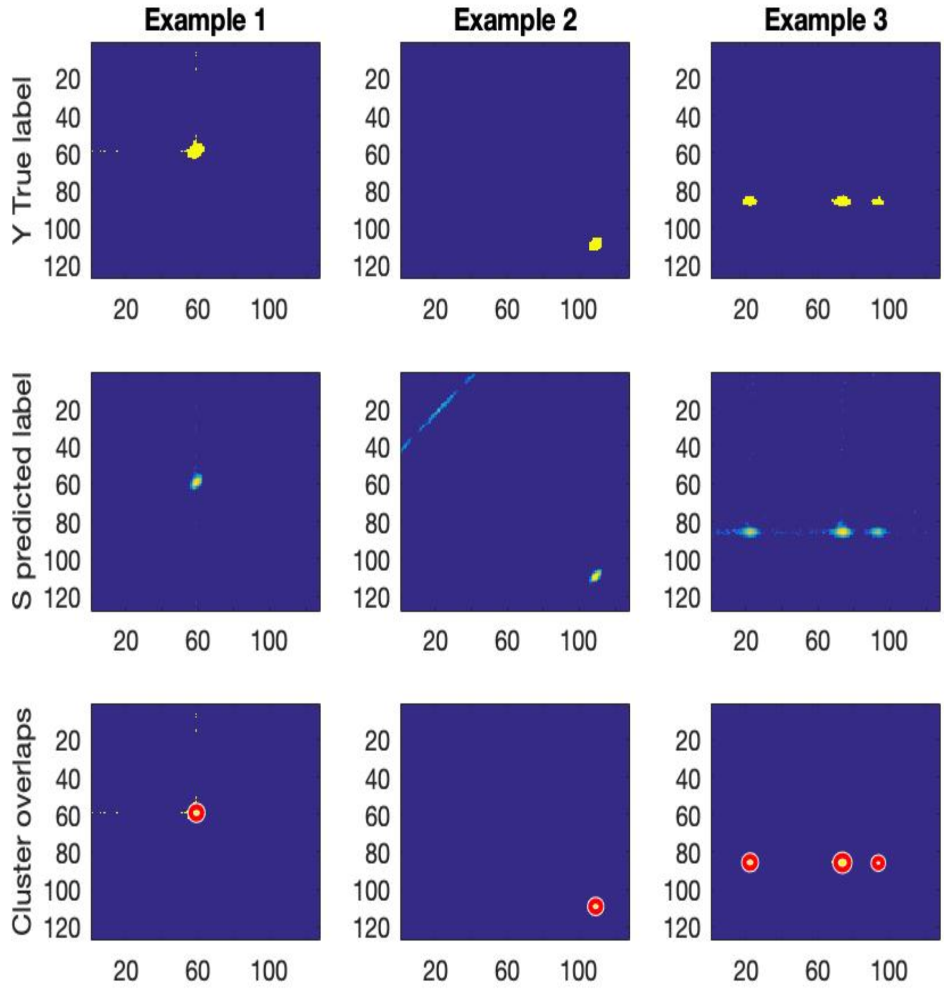


Figure 15: Comparison between true labels Y, S (foreground, predicted labels) and clusters defined on S overlaid on Y at threshold 0.4

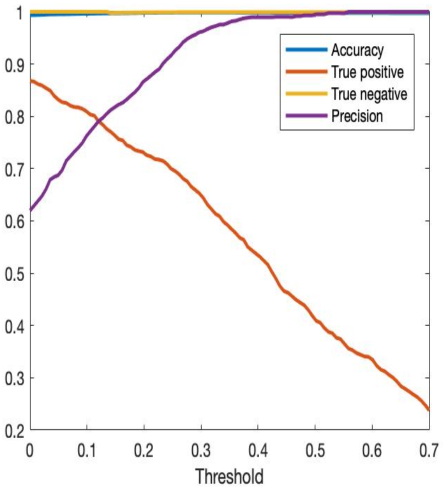

1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Threshold
Accuracy
True positive
True negative
Precision

Figure 16

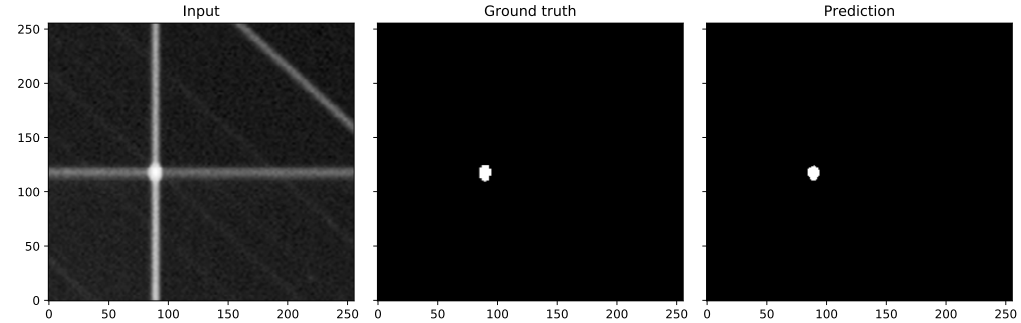


Figure 17: Example of the prediction quality of U-net. The figures at the left, center, and right show the input, training label (ground truth), and prediction by U-net, respectively.