Accelerated Change Detection in Synthetic Aperture Radar Images Based on Deep Neural Networks

Richard Anderson, Katie Snyder*, Yuanwei Jin⁺ Enyue Lu**

**Department of Mathematics & Computer Science, Salisbury University, Salisbury, MD;*

⁺Department of Engineering and Aviation Sciences, University of Maryland Eastern Shore, Princess Anne, MD

Abstract

Change detection is to identify changes between two images that are taken at different times over a specific area. It is an important problem for both civil and military applications. Change detection for Synthetic Aperture Radar (SAR) images are often difficult because the SAR images consists of an abundance of speckle noise. Current change detection methods involve generating a Difference Image (DI) and analyzing the DI. This project attempts to apply the concept of deep neural networks (DNN) to detect changes between two images, avoiding the process of analyzing a DI and/or proactively reducing noise. Our method is composed of 3 steps: filtering and preclassifying before and after SAR images to obtain good samples to train the neural network, creating and training networks, and analyzing results of the network. We demonstrate the effectiveness of our methods by conducting experiments on synthetic images and real-world NASA images.

Deep Neural Network

A DNN is a mathematical model to represent feature recognition. The neural network consists of a network of nodes in layers, where certain nodes are connected. These connections have different weights and these nodes have biases. An activation of a node can, in turn, activate a connected node based on the following function:

σ represents the logistic function

Wi represents the weight of the connection

vi represents the state of the input node

ci represents the bias of the connection

$$\sigma(W_i v_i + c_i)$$

The weights of the connections are initially set randomly. The input layer of nodes are set as the features of the good sample neighborhoods. After updating the states of all nodes in the network, the neural network reconstructs a set of input states based on the states of the output node. The weights are then updated based on the following function:

v_i represents input state

h_j represents output state

ε represents a chosen learning rate

$$\varepsilon(v_i h_{j\ initial} - v_i h_{j\ reconstructed})$$

We trained a restricted Boltzmann machine network (RBM), which consists of a type of layer-by-layer training that restricts nodes from communicating in their own layer.

Filtering Methods

To optimize the ability to detect change using the network output we looked into filtering out speckle noise, a natural occurrence in Synthetic Aperture Radar images, as a preliminary function for the image to streamline edge detection.

1. Frost Filter

Performs the filtering with local statistics computed based on neighboring pixels as a specified in the logical valued matrix MASK.

$$\mathbf{DN} = \sum_{n\,x\,n} k\alpha e^{-\alpha|t|}$$

Where α is $(4/n\sigma'^2)(\sigma^2/\bar{I}^2)$, k is the normalization constant, $\bar{I}$ is the Local Mean, σ is the Local Variance, σ' is the Image coefficient of variation value, $|t| = |X\text{-}X_0|+|Y\text{-}Y_0|$ and n is the moving kernel size

2. Lee Filter

Lee Filter computes a linear combination of the center pixel intensity in a filter window with an average intensity of the window for removing speckle noise. This filter is based on the minimum mean square error (MMSE), and speckle free image is produces based on the following equation:

$$R(t) = \bar{I(t)} + W(t)(I(t) - \bar{I(t)})$$

where $R(t)$ is image value after being filtered and it's also the estimated value of $R(t)$, $\bar{I(t)}$ is the mean value of $I(t)$, and the weighting function $W(t)$ is given by

$$W(t) = 1 - \frac{C_v}{C_I}$$

where C_v is the variance coefficient of noise-affected image with standard deviation, and C_I is the variance coefficient of noise-free of local image with standard deviation.

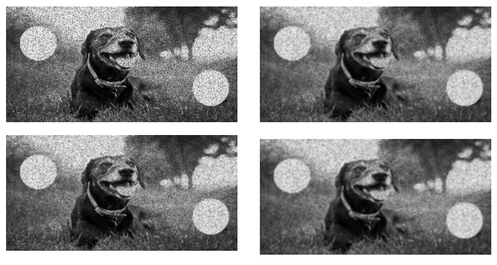
For our project, we used the image processing toolbox to filter out speckle noise with a window (neighborhood) size of 3x3 pixels.

3. Mean Filter

This filter works by calculating the mean value of the pixels of neighbor window. Then the mean was used to replace the center pixel value window. This filter noise smoothing ability is good, but it also causes a reduction of detail and resolution. This filter is can be implemented based on the formula below:

$$f(x,y) = \frac{1}{m\,.\,n}\sum_{(s,t)\,\epsilon\,Sxy} g(s,t)$$

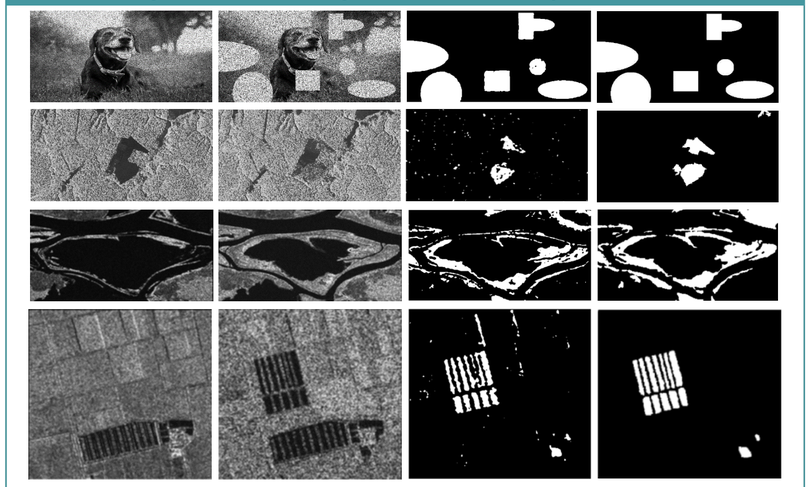
where m.n is kernel windows size, g(s,t) given original image, S_{xy} represent the set of coordinates in rectangular image window



a) 0.20 Speckle Variance b) Frost Filter c) Lee Filter d) Mean Filter

Experiments

To empirically test the effectiveness of noise filtering on the network's ability to detect changes, we ran a series of experiments on both synthetic images and real SAR images. For the synthetic images, we used Matlab's image processing toolbox to add various levels of speckle noise onto the images. We then measured how the network performed with and without filtering for varying levels of noise. For real, unaltered SAR images we ran the network four times: once for each of the three different filtering methods and once without filtering. We found that the performance of the network improved after filtering in every case we measured for both synthetic and real SAR images.



a) Test Image b) Santarem, Brazil c) Estuary in Ottawa, Canada d) Field in Ottawa, Canada

** All images are pre-filtered using the Mean Filter before becoming input for Neural Net

Optimization of the Network

To generate a proper change map, one must define what it means for a pixel to be "changed". Because 'changed' is a subjective term, there must be a given degree of change, or threshold of lightness and darkness values which the pixel must fall under to be considered different or not. Each change value is between 0 (darker) and 1 (lighter). Every single pixel is checked to see if it is within the threshold range. If the feature is not within the range it is mapped with a 1 to show it has changed or to 0 to show it is the same. Throughout our research, we found that the threshold values were crucial to generating a proper change map. Also, the "best" threshold values varied from image-to-image.

Neural Network Quantitative Results

Our experiments currently use two different equations to measure how accurate the network results are.

1. Percentage Correct Classification (PCC)

The measurement of correctly classified pixels divided by the total number of pixels in the image implemented in the equation below:

$$\mathbf{PCC} = \frac{\mathbf{TP} + \mathbf{TN}}{\mathbf{TP} + \mathbf{TN} + \mathbf{FP} + \mathbf{FN}}$$

True Positive – pixels correctly classified as changed
True Negative – pixels correctly classified as unchanged
False Positive – pixels incorrectly classified as changed
False Negative – pixels incorrectly classified as unchanged

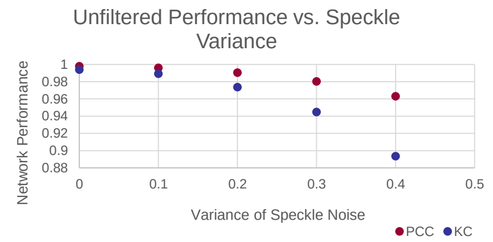
2. Kappa Coefficient (KC)

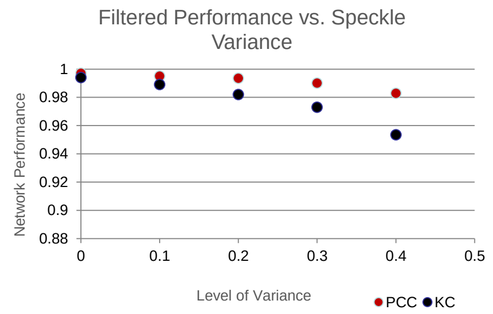
The Kappa Coefficient is a statistic which objectively measures the predictive ability of a model. It is generally thought to be a more reliable measure than simple PCC, since it takes into account the possibility of the agreement occurring by chance. If the change detection result and the reference image are in full agreement, the Kappa value is 1. Likewise, if there is no agreement the Kappa value is 0. The formula is as follows:

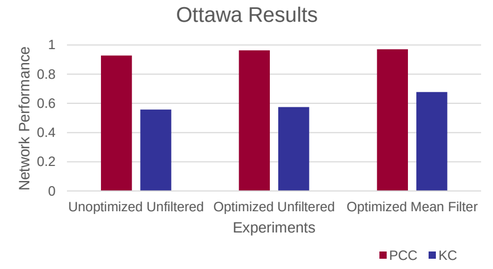
$$Kappa = \frac{PCC - PRE}{1 - PRE}$$

where PRE = $\big((TP + FP)(TP + FN) + (FN + TN)(TN + FP)\big)/(TP + TN + FP + TN)^2$

Shown in the next column are the results from some of our various experiments. There is a noticeable increase in change detection when the speckle noise is filtered out of the Before and After images before they are used as input for the network. There is also a comparison between the Kappa and PCC values for each output.







Future Work

Future work for this project includes gathering more data in order to have more confidence in the experimental results and make sure that they are statistically significant. On top of this, it would also be useful to establish a set of cases in which each different filtering method would have the best results. Finally, implementing a new kind of DBN would also be useful because different types of DBNs could potentially perform better.

References

F. Liao, E. Koshelev, M. Milton, Y. Jin, and E. Lu, "Change Detection by Deep Neural Networks for Synthetic Aperture Radar Images," Research Experience for Undergraduates, August 2016.

M. Gong, J. Zhao, J. Liu, Q. Miao, and L. Jiao, "Change detection in synthetic aperture radar images based on deep neural networks," IEEE Trans. Neural Networks and Learning Systems. vol. 27, no. 1, pp. 125-138, January 2016.

Mascarenhas, N. D. (1997). An Overview of Speckle Noise Filtering in SAR Images. IEEE, 71-79. Retrieved June & July, 2017.