Towards Detecting Stealthy Attacks in Power Grid using Deep Learning

Mohammad Ashrafuzzaman, Yacine Chakhchoukh and Frederick T. Sheldon
Departments of Computer Science and Electrical & Computer Engineering, University of Idaho, Moscow
Stealthy Data Integrity Attacks

- Surreptitiously changing data
- Intelligently and incognito
- Fooling the SCADA operators
- Cumulative ripple effect can be disastrous
Insider Threat
Outside Attacker
Stealthy Attacks in Power Grid

• Get access to one or more SCADA control Centers (in a Substation)
• Modify actual measurement data to deceive operators

Detection Mechanism:
• Find anomalous data pattern
Statistical and Machine Learning Approaches

• Statistical Methods
  • Weighted Least Squares
  • Least Trimmed Squares
  • Chi Squares
  • And more

• Machine Learning Methods
  • Distance Ratio Estimator
  • K-Nearest Neighbor
  • Support vector Machines
  • And more
Deep Learning Based Approach

- Deep Learning is being used for predictive analytics and anomaly detection in many different and diverse areas.
- Why not then to detect bad data in power grid!
So Many Deep Learning Methods

- Stacked Auto-Encoder
- Deep Belief Network
- Deep/Restricted Boltzmann Machine
- Convolutional Neural Network
- Recurrent Neural Network
- And many more!!

Each of these have variations on the theme.
Preprocessing

• Need to pre-process data before applying deep learning method
• For example: For selecting appropriate predictors or features
So Many Methods Again

- Random Forest Classifier or Regressor
- Principal Component Analysis (PCA)
- Quadratic Discriminant Analysis (QDA)
- Regularized Discriminant Analysis (RDA)
- Linear Discriminant Analysis (LDA)
- Even, unsupervised deep learning
More Variations

• Each of these methods can further be fine-tuned and optimized by varying the hyper-parameter values
How to Measure

- Use Confusion Matrix

<table>
<thead>
<tr>
<th>Total=(n)</th>
<th>Predicted Normal</th>
<th>Predicted Attack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Normal</td>
<td>TN</td>
<td>FP</td>
</tr>
<tr>
<td>Actual Attack</td>
<td>FN</td>
<td>TP</td>
</tr>
</tbody>
</table>
How to Measure

• Metrics to Evaluate
  • Accuracy \[\frac{(TP+TN)}{Total}\]
  • Precision \[\frac{TP}{(FP+TP)/Total}\]
  • Recall \[\frac{TP}{(FN+TP)/Total}\], aka, Detection rate
  • False Positive Rate \[\frac{FP}{(FP+TN)/Total}\]
  • Misclassification Rate \[\frac{(FP+FN)}{Total}\]
  • Specificity \[\frac{TN}{(TN+FP)}\]
  • Prevalence \[\frac{(FP+TN)}{Total}\]

• Execution Time
  • Time for Training
  • Time for real-time detection
The Matrix

• Perform an experiment with
  • a feature selection method
  • a deep learning method
  • A set of hyper-parameter values

• Tabulate the performance metrics
• Repeat with changing one of the three above

Will yield a comparison matrix
IEEE 14-Bus System
Data Set

• Power Grid SCADA dataset:
  • 40 active power-flows
  • 14 active power-injections and
  • 68 reactive power and voltage measurements.

• 10,000 sets of measurement data
• 1 bus is compromised
• Attack simulated by randomly modifying data at slack Bus
Feature Selection

• Random Forest Classifier
Anomaly Detection

- Stacked Autoencoder
  - Feedforward
  - 4 hidden layers
  - 50 hidden cells in each hidden layer
  - Tanh activation function
  - 50 epochs
  - 0.005 learning rate
  - 70%-30% train-test split
## Performance Matrix

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy [(TP+TN)/Total]</td>
<td>93%</td>
</tr>
<tr>
<td>Recall [TP/(FN+TP)/Total]</td>
<td>84%</td>
</tr>
<tr>
<td>Precision [TP/(FP+TP)/Total]</td>
<td>47%</td>
</tr>
<tr>
<td>False Positive Rate [FP/(FP+TN)/Total]</td>
<td>5.8%</td>
</tr>
<tr>
<td>Misclassification Rate [(FP+FN)/Total]</td>
<td>6.9%</td>
</tr>
<tr>
<td>Specificity [TN/(TN+FP)]</td>
<td>94%</td>
</tr>
<tr>
<td>Prevalence [(FP+TN)/Total]</td>
<td>90%</td>
</tr>
</tbody>
</table>