Semi-supervised EM & Weak-Supervision in Anomaly Detection

Minakshi Mukherjee; Suvasis Mukherjee; {adaboost,suvasism}@stanford.edu

Overview

Anomaly Detection from an unlabeled high dimensional dataset is a challenge in an unsupervised setup. We analyzed NSL-KDD dataset, an improved version of UCI archive KDDCUP'99 network anomaly dataset using unsupervised and semi-supervised Gaussian Mixture Model(GMM) and observed how the choice of hyperparameters influences accuracy of the algorithm. Our goal was to establish an AI framework for unlabeled or inadequately labeled anomaly detection dataset using semi-supervised GMM.

Mechanism:

1.Choose k-dimensional features out of n-dimension($n > k$)that captures maximum variability in the data

2) Choose a representative sample for each run

3) Vary hyperparameters and evaluate model accuracy

Data Set and Features

NSL-KDD20% data from UCI archive that has a wide variety of intrusions simulated in a military network environment, labeled with ground truth.

Training data: 25,171 connection vectors with 41 features labeled as normal or attack with 4 attack categories. Training set has 24 attack types for those 4 attack categories.

Test data: 4508 connection vectors with 41 features labeled as normal or attack with 4 attack categories. Test set has 38 attack types for those 4 attack categories.

Training and test data come from different probability distribution.

Data Selection Strategy

Our project uses unsupervised and semi-supervised EM algorithms, hence, we did not use the test data set as mentioned above. In order to understand the feature distribution, small representative data set for both training and test are selected out of 25,171 training data so that model iteration and evaluation can be deterministic. We followed the attached scheme for data selection:

		Simulation of Class imbalance					
	Training	Split 13449 Normal records into 5 parts	Normal1	Normal2	Normal3	Normal4	Normal5
Normal	13449		2689	2689	2689	2689	2689
Attack	11722						
Total	25171	Keep 1% of attack in each data set	Attack1	Attack2	....	....	Attack20
		Split attack into 20 equal parts	5	5	....	....	5
		Prepare 20 subsets as follows					
			Subset1	Subset2	....	....	Subset20
		Take 1 part of Normal and 1% of attack as one subset	2689 +5	2689 +5	....	....	2689 +5
		For unsupervised:Final dataset for iteration due to limited compute power on 16 GB laptop(99% of the normal data + 1% of attack data from the above splits)	1000	1000	....	....	1000
			Subset1	Subset2	....	....	Subset20
		For Semi-supervised, same distribution of data as above, but took 2% labeled data out of (1000) at random	1000	1000	....	....	1000

Models

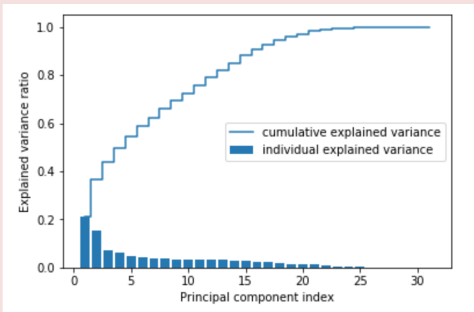
Unsupervised EM, Semi-Supervised EM

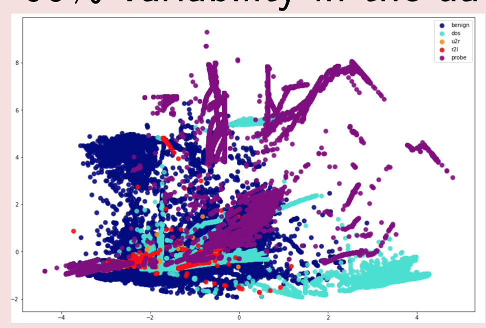
Weak Supervision

Algorithm

Feature Selection:

Due to limited compute on laptop, we used PCA of dimension 6 for feature selection that accounts for 60% variability in the data.





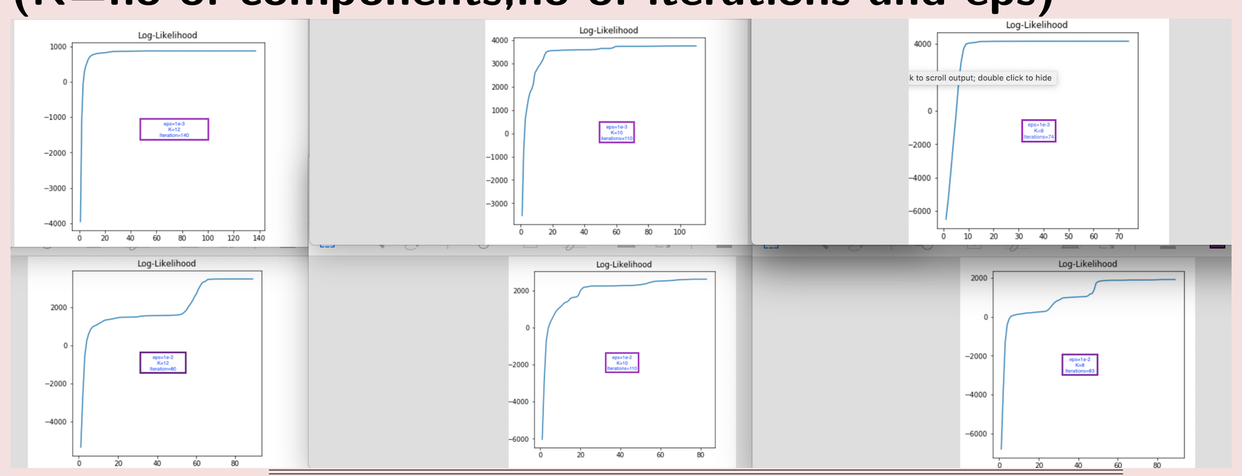
Unsupervised EM: In order to run an unsupervised EM, we remove the label column and keep that aside for ground truth comparison. Hence, we have n training sets of network connection vectors $\{x^{(1)}, \ldots, x^{(n)}\}$ with 41 features without the label column and learn the model parameters μ, Σ, ϕ through the iterative E and M step by maximizing a tractable lower bound of $p(x;\theta)$.

$$\ell_{\text{unsup}}(\theta) = \sum_{i=1}^{n} \log \sum_{z^{(i)}=1}^{k} p(x^{(i)}|z^{(i)};\mu,\Sigma)p(z^{(i)};\phi)$$

$z^{(i)}$ indicate which of the k Gaussians each $x^{(i)}$ had come from.

Few results for different hyperparameters

(K=no of components,no of iterations and eps)



Best hyperparameter for unsupervised

loglikelihood converged in 74 iterations for K=8,eps=1e-3

Semi-supervised EM

In order to run Semi-supervised EM, we take few training sets as labeled examples(as per Data Selection Strategy table layout). $\{(\tilde{x}^{(1)}, \tilde{z}^{(1)}), \ldots, (\tilde{x}^{(n2)}, \tilde{z}^{(n2)})\}$ where both x and z are observed, and remaining training sets as unlabeled. We simultaneously maximize the marginal likelihood of the parameters using the unlabeled examples, and the full likelihood of the parameters using the labeled examples. The objective function to maximize is:

$$\ell_{\text{semi-sup}}(\theta) = \ell_{\text{unsup}}(\theta) + \alpha \ell_{\text{sup}}(\theta)$$

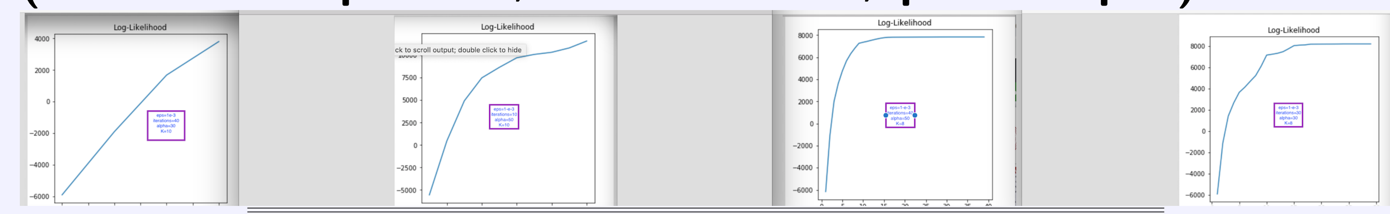
$$\mu^{(t+1)}, \Sigma^{(t+1)}, \phi^{(t+1)} := \arg\max_{\theta}$$

$$\left[\sum_{i=1}^{n_1} \left(\sum_{z_{(i)}=1}^{k} Q_i^{(t)}(z^{(i)} \log \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i^{(t)}(z^{(i)})} \right) + \alpha \left(\sum_{i=1}^{n_2} \log p(\tilde{x}^{(i)}, \tilde{z}^{(i)};\theta) \right) \right]$$

Results for Semi-Supervised EM

Few results for different hyperparameters

(K=no of components,no of iterations,eps and alpha)



Best hyperparameter for semi-supervised

loglikelihood converged in 30 iterations for K=8,eps=1e-3

Weak Supervision: Label training data using labeling api from snorkel

In order to test weak-supervision, we took complete training data of 25,177 records and removed the y columns. We used Labeling functions (LFs) to encode domain knowledge and other supervision sources programmatically –few Heuristic LFs created by us:

rules: r2l_kdd_rules.pdf Formulation of heuristic rules for R2L attacks

1. If during an FTP session, large amount is data is sent from source as compared to destination then warezmaster attack can be concluded.

   (duration > 265) ∧ (protocol = tcp) ∧ (service = ftp ∨ ftp_data) ∧ (source_bytes > 265616) ∧ (destination_bytes = 0) Warezmaster Attack
2. If a guest has logged in through an FTP connection, and hidden directories are created then warezmaster attack can be concluded.

   (protocol = tcp) ∧ (service = ftp ∨ ftp_data) ∧ (hot > 0) ∧ (hot <= 2) ∧ (is_guest_login = 1) Warezmaster Attack
3. (duration > 265) ∧ (destination_bytes <= 688) ∧ (is_guest_login = 1) Warezmaster Attack
4. If a user, during an FTP session, triggers notably many hot indicators to be set in a small duration of time then the user maybe downloading illegally posted software from the server

   (duration < 5) ∧ (protocol = tcp) ∧ (service = ftp ∨ ftp_data) ∧ (logged_in = 1 ∨ is_guest_login = 1) (hot > 25) Warezclient Attack
5. (duration <= 4685) ∧ (hot > 0) ∧ (hot <= 25) not Warezclient Attack
6. (source_bytes > 265616) ∧ (source_bytes <= 283618) Warezmaster Attack

There are 24 different attack types in the data and we handled only one Rule category and applied it on training data, by comparing the accuracy with the ground truth on training data, naturally the accuracy is much much lower as found below.

Accuracy: 21.8%

Model Accuracy and Discussion

Questions to answer: Did this GMM algorithms discover the known structure of the data? For a real-world unsupervised learning problem, this question is hard to answer, but here we compared our results against the ground truth labels.

We calculated the Detection rate and the False positive rate on all iterations and here is a summary:

	Accuracy	Detection_Rate (label_1)	Detection_Rate (label_2)	Detection_Rate (label_3)	Detection_Rate (label_4)
Unsupervised					
K=6, eps=1e-3	0.845	0.79	1	0.614	0
K=7, eps=1e-3	0.833	0.809	0.86	0	0
K=8, eps=1e-3	0.87	0.85	0.93	0.56	0
K=9, eps=1e-3	0.865	0.832	0.915	0	0
K=10, eps=1e-3	0.849	0.866	0.83	0.83	0
K=12,eps=1e-3	0.862	0.829	0.95	0.54	0
Semi-supervised					
K=8, eps=1e-3,alpha=30	0.912	0.882	0.91	0.59	0.33
K=10, eps=1e-3,alpha=30	0.89	0.91	0	0	0
K=12, eps=1e-3,alpha=30	0.898	0.892	0.83	0	0

We proved the following facts for unsupervised EM and semi-supervised EM in all iterations:

Semi-supervised EM(SSEM) converges much faster

SSEM provides stable data assignments within the mixture

Additional labels helps SSEM to uncover distribution correctly