The Thirty-First International Florida
Artificial Intelligence Research Society Conference (FLAIRS-31)

Peer Group Metadata-Informed LSTM Ensembles for Insider Threat Detection *

Jason Matterer
Jason.Matterer@ll.mit.edu
MIT Lincoln Laboratory

Daniel LeJeune
dlejeune@rice.edu
Rice University

Abstract

The problem of detecting insider threats i.e. authorized individuals who pose a threat to an organization is challenging. Since insiders have authorized access to and use sensitive data and systems on a day-to-day basis, the difference between an attack and benign normal behavior is small. We propose a method to address these issues by leveraging peer group metadata to build more robust models of normal behavior and investigate how to make use of multiple of these models and aggregate the results. Our experiments show that the use of peer group metadata improves performance over individual models trained using either hand-crafted features or event sequences.

Introduction and Related Work

The problem of detecting insider threats i.e. authorized individuals who pose a threat to an organization is challenging. In contrast to intruders, insiders, through their daily work, are often quite knowledgeable about the location of sensitive information that could both lead to potentially higher cost attacks[1] and be more difficult to detect. Since insiders have authorized access to and use sensitive data and systems on a day-to-day basis, the difference between an attack and benign normal behavior is small.

There are a number of methodologies for the insider threat problem [see (Sanzgiri and Dasgupta 2016) for a more complete taxonomy]. We focus on a data driven approach applying anomaly detection to users. A thorough survey of anomaly detection is provided in (Chandola, Banerjee, and Kumar 2009) and a survey focused on anomaly detection for sequential data is provided in (Chandola, Banerjee, and Kumar 2012).

*This material is based upon work supported by the Assistant Secretary of Defense for Research and Engineering under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Assistant Secretary of Defense for Research and Engineering.

[1]Ponemon Institute 2017 Cost of Data Breach Study https://public.dhe.ibm.com/common/ssi/ecm/se/en/sel03130wwen/SEL03130WWEN.PDF

The data used for anomaly detection based insider threat detection is typically application level logs collected from endhosts. The CERT division at the Software Engineering Institute at Carnegie Mellon University has produced a number of variants of synthetic data designed for insider threat detection (Lindauer et al. 2014; Glasser and Lindauer 2013). Since these data sets are publicly available and have no restrictions on publication, many previous methods use these data. These data sets consist of email, web, logon, file, and usb connection activities for all simulated users as well as LDAP databases providing metadata about each user. Other data sources include Active Directory logs (Hsieh et al. 2015), netflow and audit logs (Mayhew et al. 2015), text written by users (Contreras et al. 2015; Gavai et al. 2015) and psychological indicators (Contreras et al. 2015; Greitzer and Frincke 2010).

Efforts to detect or predict attacks by insiders is further complicated by high diversity of behaviors within organizations. Large organizations often consist of different departments tasked with separate missions and goals, and the background and tasks of users within their computer systems differ throughout these divisions of the organization.

The common approaches to this problem are to either build a single overarching model of behavior within the organization or build individual models for each user. However both of these approaches ignore the potentially valuable information available from how the organization is structured. The primary goal of this work is to introduce a method to leverage this "metadata" about users within an organization. By "metadata", we are referring to data about users that indicate commonality or similarity. For example, these may include the role, position, level, or job title of a user, the projects or teams a user is currently working on, or the larger group or department a user belongs to within the organization. This approach addresses two major issues user behavioral modeling for insider threat detection: (1) the cold start problem for new users, and (2) overfitting introduced from individual models.

In (Senator et al. 2013), the authors evaluated a wide variety of anomaly detection methods ranging from ensembles of Gaussian mixture models and classifier adjusted density estimation (CADE) (Hempstalk, Frank, and Witten 2008), to temporal models like particle filters (Mappus and Briscoe 2013) and graph based algorithms (Riedy and Bader 2013).

In their experiments on two months of data collected from an enterprise, CADE was the best performer using AUC and precision@k as metrics. However, the feature driven methods (like GMMs and CADE) used 111 calculated features specified beforehand by subject matter experts aggregated on a daily basis. This necessitates that any temporal interaction within a day must be encoded by feature engineering. In this work, we use the raw sequence of events so that custom feature engineering is not required and temporal dependence can be captured by the models.

The raw event sequences were used in (Rashid, Agrafiotis, and Nurse 2016) to fit hidden Markov models (HMMs) for insider threat detection. Rashid et al. proposed a system to identify anomalous user behavior on a weekly basis that was validated on the CERT data set r4.2. They fit an HMM, using randomized restarts for each user, initially trained on 5 weeks of data and retraining on new data throughout the testing period. Our work can be viewed as a natural extension of this work.

We apply Long-Short Term Memory (LSTM) networks for anomaly detection. Introduced by (Hochreiter and Schmidhuber 1997), LSTM is a type of recurrent neural network that has been applied to anomaly detection in a variety of applications. LSTMs were investigated for anomaly detection in multiple types of time series data in (Malhotra et al. 2015), which introduced a stacked LSTM model to capture long-term temporal dependence in the data. In (Taylor, Leblanc, and Japkowicz 2016), LSTMs were used to identify anomalies in the Controller Area Network (CAN) bus of vehicles. In (Cheng et al. 2016), the authors proposed a system for detecting anomalies in Border Gateway Protocol (BGP) traffic at multiple scales.

LSTMs have been previously applied to insider threat detection on the CERT data set by (Tuor et al. 2017). They use a number of count features combined with indicators of each LDAP category for a user. In their experiments, they found that removing the LDAP metadata resulted in a minor improvement to performance. In contrast, we propose building distinct models for the user metadata in LDAP and aggregating these models.

In the section titled Anomaly Detection Methods, we introduce the LSTM model and describe our approach to leveraging peer groups to build ensemble models for insider threat detection. In the section titled Empirical Results, we introduce standard baseline models to compare our results to, describe the data set we use for comparison, and describe in more detail the LSTM architectures we used and approach to ensembling.

Anomaly Detection Methods

Long-Short Term Memory Recurrent Neural Networks

Recurrent neural networks (RNN) are designed to capture long run behaviors and patterns in input data. This enables them to achieve better performance compared to feed-forward networks in a variety of applications. However, the long term dependence has a downside that training suffers from the "vanishing gradient" problem (Bengio, Simard, and Frasconi 1994). LSTMs addressed this issue via a "forget gate" where stored information about previous observations is dropped during the training process.

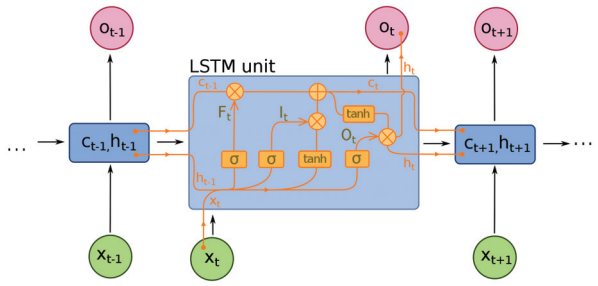

Figure 1: Depiction of a single LSTM unit including the input (I_t), output (O_t), and forget (F_t) gates.

For the insider threat detection problem, we use the raw event sequences of observations (see table 2 for a full list) one-hot encoded as the input into the models we build and the task these models are trained for is to predict the next event in the sequence. We investigate architectures with one or two layers of LSTMs with varying memory cell dimension and a final fully connected layer. This produces a generative model for the next observation of a sequence of events given the history. For model i, let $h_t^{(i)}$ denote the output of its LSTM. This is fed into a fully connected layer which produces a probability distribution $p_t^{(i)}(\cdot)$ over events.

We build two types of models, the first are trained using only the data for an individual user and the second are trained using the data from groups of users who share certain metadata. For the latter models, for each user, depending on their specific metadata, a number of the models are aggregated to calculate anomaly scores. In particular, we consider approaches that weight models based on the number of users that are used in training the model. We hypothesize that this will reduce the effect of the less useful general models, while still being robust when compared to noisy individual models.

Note that for a newly observed event the output of each RNN model is the probability under that model of observing the event. However, when comparing models trained on different data or sets of data, these probabilities are not comparable. Therefore some standardization or normalization is necessary to combine the probabilities. Based on some preliminary analysis of a random sample of users, a Gaussian distribution is fit to the scores on the training data for each model. Anomaly scores are subsequently calculated by normalizing the RNN probability using estimated mean and variance from this distribution.

Given training data events $e_1, \dots e_T$, the i^{th} generative model is used to calculate the likelihood of the observed sequence

$$p_0^{(i)}(e_1), p_1^{(i)}(e_2), \dots, p_{T-1}^{(i)}(e_T).$$

Let $\mu_T^{(i)}$ and $\sigma_T^{2(i)}$ denote the empirical mean and variance of this sequence of estimated probabilities. Then, for $t > T$, the anomaly score $\tilde{s}_t^{(i)}$ of a newly observed event e_{t+1} is

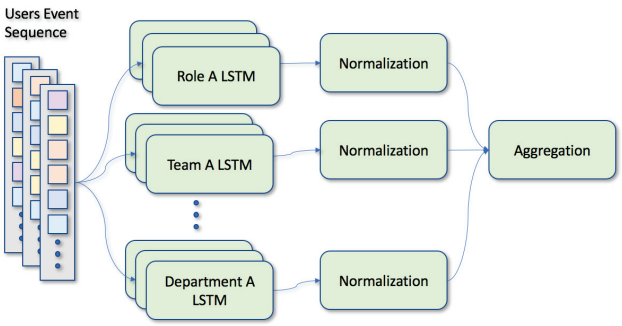

Figure 2: Overview of the user metadata group model ensembles

given by

$$\tilde{s}_t^{(i)}(e_t) := \frac{p_t^{(i)}(e_t) - \mu_T^{(i)}}{\sigma_T^{(i)}}.$$

Organization Group Ensembles

Many organizations are hierarchical and have overlapping partitions of users. We seek to leverage this user metadata to improve the performance of building normal behavior models for users.

Denote the set of users in an organization by $\mathcal{U}$, and associate to each user a collection of user metadata D_u for $u \in \mathcal{U}$, for example a job title, department they work in, projects they work on, etc. Let $\mathcal{D}$ denote the alphabet of possible user metadata.

Then for each distinct user metadata $d \in \mathcal{D}$ we can associate a set of users

$$U_d := \{u \in \mathcal{U} : d \in D_u\}$$

that have metadata d.

For each metadata d, we aggregate all the sequential data for users in U_d and train a single LSTM model M_d.

Given a new sequence of observed data for a user u, the models $\{M_d : d \in D_u\}$ are ensembled to compute a combined anomaly score. For notational simplicity, for metadata d, in superscripts we refer to model M_d by d. Then for user u, the anomaly score of the observed event e_t is given by the average

$$s_{t-1}^{(u)} := \sum_{d \in D_u} w_d \tilde{s}_{t-1}^{(d)}(e_t),$$

where the weights (w_d) may depend on the metadata group d. Specifically, we look at weights that are a function of the size of U_d, i.e. $w_d = f(|U_d|)$. We investigate a number of choices for f in the sequel.

This enables building models that capture common behaviors across similar groups of users in an organization and, in comparison to individual user models, reduces the likelihood of overfitting specific observed behaviors. Additionally, for data hungry methods like LSTMs, more data is used in the training process.

Source	Device	Email	File	HTTP	Logon
Count	13	16	12	12	11

Table 1: Feature counts

Empirical Results

To evaluate the effectiveness of LSTMs and the user metadata ensembling approach, we performed a number of experiments on one of CERT's synthetic insider threat data sets. All of the models were trained and evaluated using the Keras[2] Python package version 2.0.3 with a TensorFlow[3] (Abadi et al. 2016) version 1.3.0 backend.

In the following experiments, we present performance results in terms of receiver operator characteristic (ROC) curves as well as the summary statistic area under the ROC curve (AUC). False positive rate, unlike precision, is not sensitive to class imbalance, which is a significant problem in insider threat.

We are primarily interested in three research questions: (1) for individual user models, what effects do changes in the model's architecture have on performance? (2) for the metadata-informed models, how do different ensembling strategies affect performance? and (3) how do the metadata-informed models compare to individual user models?

Baseline Methods

As in (Tuor et al. 2017), for baseline models we use one class support vector machines (OC-SVM) and isolation forests implemented in the scikit-learn (Pedregosa et al. 2011) Python module. We collected 64 features aggregated on a daily basis. These include counts of all seen activities, first/last time of each event, and email attachment statistics. We add to these an indicator for the day of the week. For many of the users, over many days, there were no events of the type being aggregated. This produced zero counts for the count features and undefined values for others. We used a per-user median imputation strategy with an additional binary feature for each imputed value.

Since the distributions of some of the previous features are heavy tailed, we apply a percentile transformation as a preprocessing step. For each feature, we replace the raw feature value with its percentile in the training data empirical distribution.

Data Set

We use CERT's synthetic insider threat data set version r6.2 (Lindauer et al. 2014; Glasser and Lindauer 2013) for all experiments. Standard network and host events were simulated for a synthetic organization of 4000 employees, encompassing email, web, usb device, file, and logon activities. Data was generated for the timespan between January 2, 2010 and June 6, 2011 covering 516 days. Into these normal behaviors, 5 distinct attacks by insiders over 47 days were injected of a variety of types.

[2]https://keras.io

[3]https://www.tensorflow.org/

Event	Count
ConnectActivity.CONNECT	778,354
ConnectActivity.DISCONNECT	773,474
EmailActivity.SEND	3,832,466
EmailActivity.VIEW	7,162,491
FileActivity.OPEN	657,500
FileActivity.WRITE	251,126
FileActivity.COPY	725,458
FileActivity.DELETE	380,799
HTTPActivity.VISIT	115,401,178
HTTPActivity.DOWNLOAD	1,413,637
HTTPActivity.UPLOAD	210,401
LogonActivity.LOGON	1,948,933
LogonActivity.LOGOFF	1,581,352

Table 2: Counts of events in the CERT r6.2 data set

Previous versions of the CERT dataset used simulate smaller organizations and introduced more attacks. We chose the r6.2 dataset because it provided the most realistic ratio of attacks per benign user.

In all experiments, we used the first 90 days of data (note these are calendar days, not work days nor days of non-empty data) from the respective users in training each of the models. We did not perform any retraining and the remaining 426 days were used for evaluation. We chose 90 days as a reasonable compromise between time until applying the results of the model and collecting sufficient training data. In contrast, (Rashid, Agrafiotis, and Nurse 2016) started with 5 weeks of training data to build hidden Markov models (with retraining every week), however, they evaluate performance on a weekly basis, and (Tuor et al. 2017) used 418 days for training of individual user LSTMs.

Even though the anomaly detection methods we evaluate produce scores for each individual event produced by a user, we aggregate these scores over a period of 24 hours to produce an anomaly score for users on each day in the test set. For days in which a user did not have any corresponding events (e.g. weekends for many users), we assumed that such days were not anomalous, setting the anomaly score of these days to zero.

Feature Extraction

For the anomaly detection methods, we restricted the discrete observable events to type and activity. This resulted in 13 possible observables: in Table 2. Because of the imbalance in the distribution of events, with HTTPActivity.VISIT comprising over 85% of all the events, we applied instance weighting during training of all LSTMs, weighting each event type by the reciprocal of the square root of its respective count in the training data. This choice of class weights appeared to produce the best results in terms of the distribution of predicted events.

User metadata was collected from LDAP data included in the CERT data set, which included role (e.g. job category), business unit, functional unit, department, team, supervisor, and projects. We used role, functional unit, department, and team as available metadata for the group models. Business unit was constant throughout all users in the data set and the number of supervisors and projects were more numerous than the other groups, which would have resulted excessive computational costs in the training process and models that would be too specific.

Metadata	Count
Role	46
Business Unit	1
Functional Unit	11
Department	23
Team	90
Project	366
Supervisor	246

Table 3: Metadata and its counts of unique values observed in the CERT data set

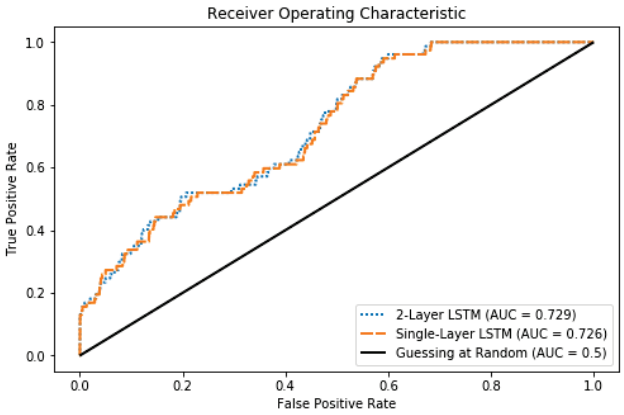

Figure 3: ROC curve comparing one and two layer LSTMs.

LSTM Architecture

The first set of experiments we performed were designed to explore the effects that tuning the architecture of the model could have on its performance as an insider threat detector. Following (Malhotra et al. 2015), we look at stacked LSTMs to capture long-term temporal structure in the data and compare the results to using a single-layer LSTM. In Figure 3, we plot ROC curves for a single and dual layer LSTMs (both with 32 memory cells). The two layer LSTM performs marginally better in terms of AUC, but from a practical standpoint, these two architectures are indistinguishable in terms of performance.

One explanation for this result is that the baseline normal behavior data is simulated by a process that may not possess much long-term temporal dependence and use in real-world data might produce different results. Another explanation is that only 90 days of events were used in the training set, in contrast (Tuor et al. 2017), who used the same data set, trained on 418 days using a total of over 111 million separate events. As more data is incorporated into the training set, the more complicated stacked model might outperform the simpler. For the remaining parameter tuning experiments we

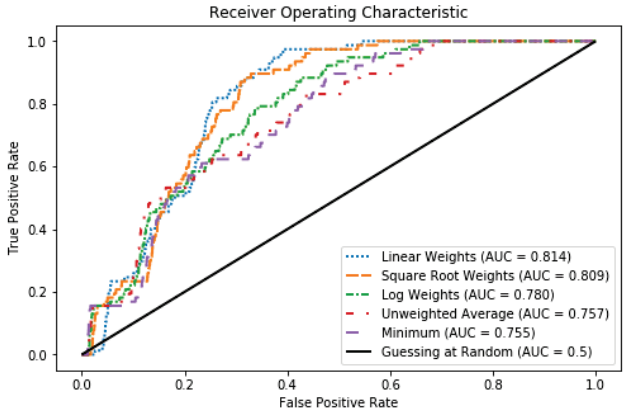

Figure 4: ROC curve of different averaging strategies of user groups

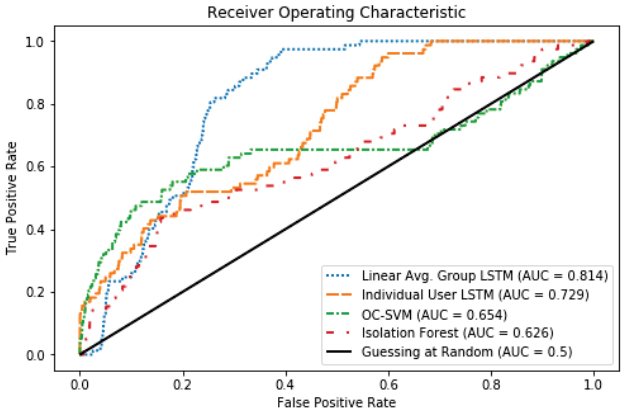

Figure 5: ROC curve comparing a two layer LSTM with 32 memory units, group metadata ensembles with unweighted and linear count weights, and the baseline one-class SVM and isolation forest algorithms.

selected the more parsimonious single-layer model.

Ensembles

We investigated a variety of aggregation strategies to produce user models from the metadata group models. Starting from the intuition that the fewer users belonging to a metadata group, the more informative it should be in predicting user's events, we look at weighted averages of the model scores depending on the number of users in the model's training set

$$f_{\text{linear}}(n) = \frac{1}{n}, \quad f_{\text{sqrt}}(n) = \frac{1}{\sqrt{n}}, \quad f_{\log}(n) = \frac{1}{\log(n+1)}.$$

In Figure 4, we show the performance of the different weighting schemes. These experiments confirm the previous intuition that inversely weighting a model's anomaly scores by the number of users belonging to that model results in a more accurate ensemble. Each of the weighting procedures—linear, square root, and log—perform better than equal weights and as the weight function becomes more sensitive to a group's size (i.e. going from natural log, to square root, to linear), AUC improves.

We can then ask what happens in the extreme extension of this strategy of penalizing large models and prioritizing smaller models. To answer this question, we consider the strategy of only using the smallest model (in terms of number of users sharing the same metadata value) for anomaly scores:

$$f_{\min}(|U_d|, u) = \mathbb{I}(d \in \operatorname*{argmin}_{d' \in D_u} |U_{d'}|).$$

However, as seen in Figure 4, this "weighting" performs slightly worse than an unweighted average. Therefore, there is useful information available in other peer groups besides the one most closely describing a user.

We compare the linear weighting and unweighted average aggregation schemes to the best performing individual user models and baseline systems in Figure 5. Over moderate regions of false positive rate (i.e. 0.3 to 0.5), the group metadata ensemble methods perform significantly better than the alternatives; however, at operating points with low false positive rate (i.e. less than 0.05), group metadata ensembled methods perform worse than random guessing. Note from Figure 4, other averaging functions than linear perform better in this region. This is most likely due to a single benign user (KDC0444), who has 93 out of the top 100 most anomalous scores according to the linear averaged method. Linear averaging disproportionately weights this user because the peer groups the user belongs to are relatively small. This user's role is an Attorney, is in the Purchasing functional unit and Contracts department (she does not belong to any team), with respective sizes 3, 18, and 6, while the average sizes of these groups are 107.6 for roles, 255.1 for functional unit, and 237.6 for department.

Conclusion and Future Work

We proposed a method for insider threat detection based on the principle that comparing a user's behavior to similar peer groups will produce a more accurate and robust system in comparison to only looking at each user individually. We have presented some empirical evidence to support the conclusion that using user metadata like a user's role, team, or department to build these models from peer groups can improve detection over individual user models. Additionally, results from these aggregated models are sensitive to how the anomaly scores are averaged.

One major avenue of future work is to examine unsupervised data driven ensembling methods on a per-user basis. The use of number of peers belonging to a group in averaging models is a crude first approach and a more refined approach that "lets the data speak for itself" is needed. Additionally, such approaches could elucidate the relative importance each peer group has on predicting future behaviors, which could, by itself, provide valuable insights for this method and user behavioral modelling in general.

In user behavioral modeling for insider threat detection, there are two major issues (1) the cold start problem for new

users, and (2) possible interference via adversarial users. User group ensembles can address both these issues. For new users without any or sufficient training data, knowledge of their metadata can be leveraged to use previously built models from similar user data, where "similar" depends on the type and availability of metadata for users. In all systems that attempt to model normal user behavior to predict or detect attacks by insiders, there is a potential attack to the system itself that a malicious user can undertake. Given a desired set of behaviors (e.g. sensitive data exfiltration via email), a malicious user can progressively modify their standard behavior (e.g. send emails with more/larger attachments of varying file types) so that when the attack is to be performed, it appears to be less anomalous with respect to what has been observed. We hypothesize that individual models trained on a single user's data are more sensitive to these manipulations, whereas models that aggregate users will be more robust; however, further work that tests such hypotheses is needed.

References

Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M.; et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Bengio, Y.; Simard, P.; and Frasconi, P. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks* 5(2):157–166.

Chandola, V.; Banerjee, A.; and Kumar, V. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41(3):15.

Chandola, V.; Banerjee, A.; and Kumar, V. 2012. Anomaly detection for discrete sequences: A survey. *IEEE Transactions on Knowledge and Data Engineering* 24(5):823–839.

Cheng, M.; Xu, Q.; Lv, J.; Liu, W.; Li, Q.; and Wang, J. 2016. Ms-lstm: A multi-scale lstm model for bgp anomaly detection. In *Network Protocols (ICNP), 2016 IEEE 24th International Conference on*, 1–6. IEEE.

Contreras, P.; Murtagh, F.; Hadlington, L.; and Scott, K. 2015. Clustering insider threat behaviour: An ultrametric anomaly detection system. In *'Proceedings of the 60th World Statistics Congress*.

Gavai, G.; Sricharan, K.; Gunning, D.; Rolleston, R.; Hanley, J.; and Singhal, M. 2015. Detecting insider threat from enterprise social and online activity data. In *Proceedings of the 7th ACM CCS international workshop on managing insider security threats*, 13–20. ACM.

Glasser, J., and Lindauer, B. 2013. Bridging the gap: A pragmatic approach to generating insider threat data. In *Security and Privacy Workshops (SPW), 2013 IEEE*, 98–104. IEEE.

Greitzer, F. L., and Frincke, D. A. 2010. Combining traditional cyber security audit data with psychosocial data: towards predictive modeling for insider threat mitigation. *Insider Threats in Cyber Security* 85–113.

Hempstalk, K.; Frank, E.; and Witten, I. H. 2008. One-class classification by combining density and class probability estimation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 505–519. Springer.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Hsieh, C.-H.; Lai, C.-M.; Mao, C.-H.; Kao, T.-C.; and Lee, K.-C. 2015. Ad2: Anomaly detection on active directory log data for insider threat monitoring. In *Security Technology (ICCST), 2015 International Carnahan Conference on*, 287–292. IEEE.

Lindauer, B.; Glasser, J.; Rosen, M.; Wallnau, K. C.; and ExactData, L. 2014. Generating test data for insider threat detectors. *JoWUA* 5(2):80–94.

Malhotra, P.; Vig, L.; Shroff, G.; and Agarwal, P. 2015. Long short term memory networks for anomaly detection in time series. In *Proceedings*, 89. Presses universitaires de Louvain.

Mappus, R. L., and Briscoe, E. 2013. Layered behavioral trace modeling for threat detection. In *Intelligence and Security Informatics (ISI), 2013 IEEE International Conference on*, 173–175. IEEE.

Mayhew, M.; Atighetchi, M.; Adler, A.; and Greenstadt, R. 2015. Use of machine learning in big data analytics for insider threat detection. In *Military Communications Conference, MILCOM 2015-2015 IEEE*, 915–922. IEEE.

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

Rashid, T.; Agrafiotis, I.; and Nurse, J. R. 2016. A new take on detecting insider threats: Exploring the use of hidden markov models. In *Proceedings of the 2016 International Workshop on Managing Insider Security Threats*, 47–56. ACM.

Riedy, J., and Bader, D. A. 2013. Multithreaded community monitoring for massive streaming graph data. In *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2013 IEEE 27th International*, 1646–1655. IEEE.

Sanzgiri, A., and Dasgupta, D. 2016. Classification of insider threat detection techniques. In *Proceedings of the 11th Annual Cyber and Information Security Research Conference*, 25. ACM.

Senator, T. E.; Goldberg, H. G.; Memory, A.; Young, W. T.; Rees, B.; Pierce, R.; Huang, D.; Reardon, M.; Bader, D. A.; Chow, E.; et al. 2013. Detecting insider threats in a real corporate database of computer usage activity. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1393–1401. ACM.

Taylor, A.; Leblanc, S.; and Japkowicz, N. 2016. Anomaly detection in automobile control network data with long short-term memory networks. In *Data Science and Advanced Analytics (DSAA), 2016 IEEE International Conference on*, 130–139. IEEE.

Tuor, A.; Kaplan, S.; Hutchinson, B.; Nichols, N.; and Robinson, S. 2017. Deep learning for unsupervised insider threat detection in structured cybersecurity data streams. In *Artificial Intelligence for Cybersecurity Workshop at AAAI*.