ReportIT: Improving Insider Threat Detection Model Explainability Through Report Retrieval

Stanford CS224N {Custom} Project
TA mentor: Gaurab Banerjee
External Collaborators, Mentors, Others Courses Sharing Project: N/A

Sameer Khanna
Department of Computer Science
Stanford University
sameerk@cs.stanford.edu

Abstract

Insider Threats are costly, necessitating models designed to detect these threats before they cause damage. Unfortunately, models that lead to high performance on benchmark datasets tend to use image encodings of behavior, which makes it unclear to a security expert the reason why the behavior is classified as malicious. Additionally, it is hard to get high quality Insider Threat data to train detection models. To solve this issue, we introduce ReportIT, a contrastive learning model that takes behavior images and generates a report detailing the user's behavior while also increasing the data efficiency of such approaches.

1 Introduction

As we move further into an ever more digital age, newer and more complex attack vectors are appearing everyday. Insiders pose a unique threat to corporations and organizations of all scales due to their access to proprietary systems and their ability to circumvent security protocols and blind spots the public is not privy to. Close to 30% of confirmed breaches today involve insiders [1]. In total, over 2,560 internal security breaches occur in United States businesses every day [2] with a year-over-year increase in insider attack rates of 21.4% [3]. Each such attack costs an organization on average 11.45 million USD annually [4].

Unfortunately, these attacks are extremely difficult to detect from within. Third party entities detect the vast majority of most data breaches that occur within an organization; famous examples include the breaches in TJX Companies, VeriSign, Adobe and LinkedIn [5, 6, 7, 8]. These failures can be attributed to the simplicity of insider threat detection systems used in industry today. While solutions proposed by academia boast higher predictive power, interpretability concerns prevent their usage as industry models. Current trends of advancement in the space of insider threat detection revolve around the usage of image encodings to represent employee behavior, leading to state of the art performance in terms of accuracy as well as false positive rate at the cost of further reducing interpretability of models [9]. As insider threat detection models can influence whether an employee gets to retain his/her job, it is imperative that the security expert assessing the situation understands the reason behind a model labeling an employee's behavior as malicious.

Additionally, there are concerns regarding data availability for insider threat detection. Traditional UEBA training data is composed of real-life scenarios, consisting of confidential information for a company, as well as the personal information of their employees. Thus, each vendor utilizes their own private datasets, making model comparisons and benchmarking difficult in nature. It is important to determine if there are methods that can make insider threat detection models more label efficient.

To this end, we developed ReportIT, a model that takes the behavior image encodings that lead to great prediction performance and generates (or retrieves) a report detailing in plain English the reasoning behind a model's classification. We also wish to evaluate if using insider threat reports can improve the label efficiency of of image encoders, increasing the ease of setup and the effectiveness of deployed insider threat detection systems in industry.

This work is novel for a couple of reasons:

- This is the first time contrastive learning has been proposed for insider threat detection.
- To our knowledge, this is the first time image-text contrastive learning has been proposed to be used for image encodings as opposed to true images like natural images or medical images.
- To our knowledge, this is the first time image-text contrastive learning has been proposed for such a highly imbalanced data space, especially one where we there is far greater diversity in the minority class than the majority class.
- We introduce two novel methods of training for multimodal contrastive learning when there is greater diversity in examples for one modality than the other.

2 Related Works

Academia Researchers have worked on a plethora of solutions for insider threat detection, the vast majority of which utilize machine learning [10]. The most popular approach is framing insider threat detection as an anomaly detection problem [11]. Chandola et al. [12] provide a detailed overview of the state of anomaly detection. Machine learning methods have been shown to effectively handle the anomaly detection problem; there is a plethora of research that has been published regarding this avenue [13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 9]. Section A goes over the current approaches proposed for insider threat detection in greater detail.

Despite academia on the topic of insider detection dating as far back as 1987 [25], very little has reached the industry. Gates et al. cites that the reason for this failure is academia tends to assume that the definition of malicious activity is universal and that understanding why an insider threat is classified as malicious is trivial, which can be far from the truth [26].

Industry Thus, while numerous User and Entity Behavior Analytics systems (UEBA), systems built in the security industry to detect insider threats, have been created, most rely on their own datasets and experience rather than using discoveries found in academia. Examples of industry implementations include Niara, which utilizes Principle Component Analysis (PCA) [27] in combination with Mahalanobis distance [28] outlier detection (MD) in their implementation. This approach has a multitude of issues; Niara claims it is highly sensitive to outliers within their training data and cannot be effectively tuned to reduce the number of false negatives predicted [29]. In their CISO guide, Aruba Networks proposes a variety of different model types and scenarios, including SVM, Naïve Bayes (NB), and Logistic Regression (LR) [30]. Fortinet utilizes NB to categorize activity by an anomaly score; the higher the score, the more likely a given user is acting in a malicious manner [31]. Exabeam utilizes a second order Factorization Machine [32, 33] with a Markov Chain Monte Carlo learning method [34] in order to improve first-time access malicious activity reporting [35].

Unfortunately, while there are numerous UEBA systems to be found in industry, most are rudimentary in nature. Bussa et al. in Gartner's market analysis report for UEBA states that most vendors still rely at least partially on rule based implementations and require upwards of half a year worth of tuning in order to achieve effectiveness [36]. There is a lot of skepticism among security practitioners regarding UEBA systems due to poor interpretability, and very few real-world deployments leverage machine learning [29, 26, 37, 38, 39].

Contrastive Learning Several approaches have been considered to improve the label efficiency of models. A common approach is to utilize transfer learning, which focuses on gathering knowledge by solving one problem and applying it to a related problem in the same domain. Transferring model weights pretrained on natural image datasets like ImageNet has been effective, leading to statistically significant boosts to performance when compared to models without pretraining [40, 41, 42].

More recently, natural image transfer learning approaches have been augmented by the use of self-supervised learning techniques that do not require explicit labels for image interpretation to create a model with high performance, further reducing the need for labeled data while still retaining high performance [43, 44]. Contrastive learning is one form of self-supervised learning that aims to create general representations of data by contrasting similar and dissimilar pairs of examples. There are many instances where contrastive learning has been utilized to improve the efficacy of deep learning models [45, 46, 47]. Implementations of image-text contrastive learning where natural language is used to guide the learning of image encoders without the use of labeled data has seen wide success across a variety of problem spaces [48, 49].

Image Captioning The traditional method for obtaining text information describing an image is via treating the task as an image captioning problem. Image Captioning is the process of generating textual description of an image. The task of image captioning can be divided into two modules. The first module, the computer vision encoder, extracts the features and nuances out of input images. The second module, the language based decoder, translates the features and objects given by our image based model to a natural language sentence. A variety of different encoder and decoder architectures have been utilized for the image captioning task with great success [50, 51, 52, 53, 54, 55, 56, 57].

3 Approach

Main Approach We develop ReportIT, a retrieval-based insider threat report generation method that uses contrastive language image pre-training (CLIP) [48]. Taking advantage of the fact that the current state-of-the-art in insider threat detection utilizes color-based images that encode the behavior of the given employee to be assessed, we use learned behavior image - report pair representations that enable our model to retrieve unstructured free-text insider threat reports. By framing the report generation problem as a retrieval task, we can take advantage of the limited space of possible reasons the behavior of an employee can be considered malicious.

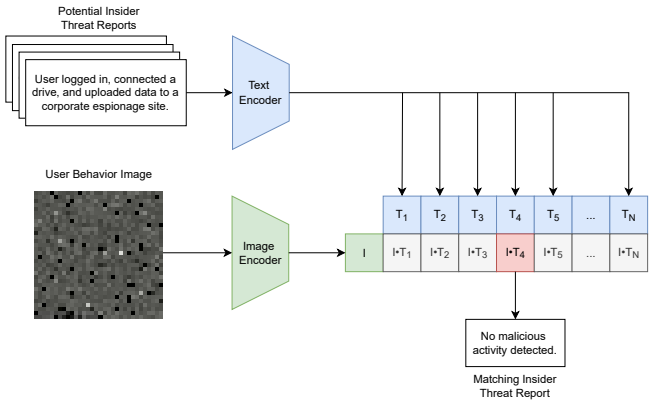

Figure 1: Details regarding how ReportIT works. Here, the given behavior images as well as the potential reports are encoded by the image encoder and the text encoder respectively. For each image encoding, we determine the corresponding report encoding that has the highest cosine similarity. The given report is returned as the retrieved report.

The general approach for ReportIT is detailed in Figure 1. Here, the given behavior image as well as the potential reports are encoded by the image encoder and the text encoder respectively, leading to the image encoding I and the report encodings $T_1, T_2, ..., T_N$. For each image encoding, we determine the corresponding report encoding that has the highest cosine similarity, as shown in Equation 1. The given report is returned as the retrieved report.

$$\text{Index of Retrieved Report} = \arg\max_i \frac{I \cdot T_i}{|I||T_i|} \tag{1}$$

Training We seek to train ReportIT such that true image-text pairs have a high cosine similarity while false image-text pairs have a low cosine similarity. At training time, we sample a batch of N input pairs (x_{image}, x_{text}), where x_{image} refers to the image and x_{text} refers to the corresponding text. Using the image encoder and text encoders, we create the subsequent encodings (I, T). We denote pair i of encodings as (I_i, T_i). Our training objective is composed of two loss functions: Equation 2 showcases the image-to-text contrastive loss for the i-th pair, while Equation 3 showcases the text-to-image contrastive loss for the i-th pair. We combine these two loss function via a simple average, leading to our contrastive loss for the training batch shown in Equation 4.

$$(loss_{image-text})_i = -\log \left(\frac{\exp\left(\frac{I_i \cdot T_i}{|I_i||T_i|}\right)}{\sum_{k=1}^{N} \exp\left(\frac{I_i \cdot T_k}{|I_i||T_k|}\right)} \right) \tag{2}$$

$$(loss_{text-image})_i = -\log\left(\frac{\exp\left(\frac{I_i \cdot T_i}{|I_i||T_i|}\right)}{\sum_{k=1}^{N}\exp\left(\frac{I_k \cdot T_i}{|I_k||T_i|}\right)}\right) \tag{3}$$

$$\mathcal{L} = \frac{1}{2N}\sum_{i=1}^{N}(loss_{image-text})_i + (loss_{text-image})_i \tag{4}$$

Issue with Traditional Contrastive Learning As detailed in Section 4, the insider threat problem space is highly imbalanced. Additionally, we want to explain why a given behavior image corresponds to malicious behavior; if the behavior image corresponds to benign behavior the generated report need not provide additional information. As a result, the majority class for the problem space has far lower diversity of possible reports than the minority class. Figure 2 details the issues with using a standard contrastive learning training regime for this problem. As there will be numerous false negative image-text pairs within a batch, contrastive learning will try to increase the distance between images and reports it should be aiming to bring close together, negatively hampering retrieval performance of the trained image and text encoders.

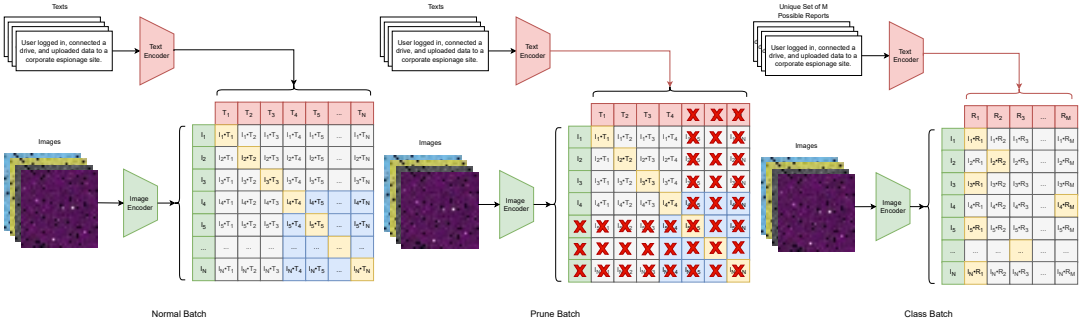

Figure 2: Various training methodologies used for ReportIT. Here, yellow squares indicate true positive image-text pairs, white/light-grey squares indicate true negative image-text pairs, and blue squares indicate false positive pairs. For the NormalBatch and PruneBatch examples, there are multiple correct reports that are in the same training batch ($T_4 = T_5 = ... = T_N$). As a result, contrastive learning will treat all image-text pairs (I_j, T_k) where $j \neq k$, $j \in [4, 5, .., N]$, $k \in [4, 5, .., N]$ as negative image-text pairs despite in actuality being positive image text pairs. PruneBatch handles this issue by pruning images and texts that would lead to this issue, while ClassBatch instead treats texts akin to a class.

Proposed Solutions Thus, we propose PruneBatch, which is detailed in Figure 2. By removing image-text pairs where there is an identical report already in the batch we reduce the false negative image-text pairs within a batch, leading to drastically improved model training and superior text retrieval using the trained model. While PruneBatch does not run into the issues a normal batching paradigm has with false negative pairs, each training batch we are essentially removing significant amounts of data that could prove to be useful in training our system. As reducing the amount of training data leads to a reduction in diversity in training examples, the subsequent model will be far more brittle and will overfit the data more easily. More importantly, it has been shown that increasing batch sizes is important for improving contrastive learning models as larger batch sizes increase the ratio of negative pairs to positive pairs. For a given batch size B, there will be $B^2 - B$ negative pairs and B positive pairs; as B increases, the number of negative pairs increases faster than the number of positive pairs. Higher negative pair to positive pair ratios have empirically lead to higher quality models [48]. With PruneBatch we effectively are doing the reverse, decreasing this ratio.

As a result, we also propose ClassBatch, which is detailed in Figure 2; note now that a single text report can be used for the correct image-text pair multiple times within a batch. Rather than treat image-text within a batch as pairs, we treat the text related to a given image as a class, where the class number corresponds to the index of the given report within the set of all possible reports. This approach provides two major benefits over PruneBatch. First, we are no longer removing training data at training time, reducing the likelihood of overfitting. Second, we are now contrasting the correct text report for a given behavior image to all possible text reports, significantly improving alignment between an image

and its given text report by maximizing the ratio of negative pairs to positive pairs irrespective of the training batch size.

Due to the highly imbalanced nature of the problem space, some reports will appear as the correct report significantly more often than others. To combat the potential issues that may occur because of this, we utilize a modified contrastive loss that takes into account class weights to train the ClassBatch approach; the new loss function is shown in Equation 5. Here, W_{T_i} denotes the weight corresponding the to the text T_i; each W_{T_i} is determined such that every report type has equal weighting during training. $R_{R==T_i}$ corresponds to the report in the set of all possible reports that matches input text T_i. Note that there is now an unequal number of images and texts being compared. Every image has only one possible report associated with it, but not every report has a single behavior image associated with it. As a result, the ClassBatch loss function is composed solely of the image-to-text contrastive loss as applied between the images in the batch and all possible reports.

$$\mathcal{L}_{ClassBatch} = -\frac{1}{N}\sum_{i=1}^{N} W_{T_i} \log \left(\frac{\exp\left(\frac{I_i \cdot R_{R==T_i}}{|I_i||R_{R==T_i}|}\right)}{\sum_{r=1}^{R} \exp\left(\frac{I_i \cdot R_r}{|I_i||R_r|}\right)} \right) \tag{5}$$

Baselines As we wish to identify the benefits of pursuing this task as a report retrieval problem as opposed to an image captioning problem, our baselines consist of a variety of image captioning models. The task of image captioning can be divided into two modules. The first module, the computer vision encoder, extracts the features and nuances out of input images. The second module, the language based decoder, translates the features and objects given by our image based model to a natural language sentence. For our baselines, we utilize a state-of-art computer vision encoder and state-of-art natural language processing decoder models using pretrained checkpoints from HuggingFace [58]. Studies have shown the effectiveness of initializing encoder/decoder models with pretrained checkpoints [50, 51, 52, 53, 54], making such models ideal for a baseline comparison of our approach. We utilize the Vision Image Transformer (ViT) as our image encoder [59], and evaluate BERT, BART, GPT-2, and RoBERTa as text decoders [60, 61, 62, 63].

4 Experiments

Data Traditional UEBA training data is composed of real-life scenarios, consisting of confidential information for a company and the personal information of their employees. Thus, each vendor utilizes their own private datasets, making model comparisons and benchmarking difficult in nature. The CERT Insider Threat center together with ExactData LLC analyzed 1,154 actual insider incidents in the United States to create the largest public repository of insider threat scenarios in order to tackle this issue [64]. Many publications and companies have utilized this dataset to assess model architectures, perform integration testing, and run confirmatory hypothesis testing, solidifying its status as the gold standard public dataset for insider threat detection system benchmarking. [65].

The CERT Insider Threat dataset contains 32,770,224 unique events, with available audit data sources including logon activity, email traffic, web browsing traces, file access logs, thumb drive usage, as well as LDAP information describing the organization hierarchy and user roles.

Malicious users within the dataset execute activities in highly variable time periods, with some attacks completing within a day, while others occur over a 2 month span. The high diversity in attack time frames enables robust verification checks against insiders that intentionally act slowly.

As occurs in the real world, this dataset is extremely imbalanced; Table 1 details the occurrence of attack scenarios compared to normal behavior within 24 hour time frames per user.

Table 1: Imbalance Ratios Within Insider Threat Data

Class Type	Number of Instances	Imbalance Ratio
Normal	330452	1 : 1
Scenario 1	85	1 : 3899
Scenario 2	861	1 : 384
Scenario 3	20	1 : 16570

Attack Scenarios There are three scenarios of attack within the dataset. In Scenario 1, a user obtains sensitive information they subsequently upload to Wikileaks. In Scenario 2, a user browses job sites

looking for a job, stealing confidential information and leaving as soon as they find one. Finally, in Scenario 3 a system administrator grows to dislike their job and downloads and installs a keylogger onto their supervisor's computer. Using the obtained password, they send an alarming mass email acting as the supervisor, leaving the organization immediately afterwards.

Evaluation Method We have two sets of experiments for ReportIT; one set evaluates the label efficiency of its trained image encoder, while the other evaluates its effectiveness in the report generation task. For the label efficiency task, we employ three experiments. First, we seek to evaluate the quality of image encoding qualitatively, using t-distributed stochastic neighbor embedding (t-SNE) [66], a dimensionality reduction technique developed for effective high-dimensional data visualization, to see how well various image encoders separate benign behavior from malicious behavior. For quantitative label efficiency evaluation, we report balanced accuracy, precision, recall, F1 score, and AUC as these metrics allow for a good evaluation of models in problem spaces with high data imbalance [67]. We use these metrics to evaluate our architectures under linear evaluation, where a linear classifier is fitted on top of a model's image encoder features to make predictions, and finetuning, where a softmax layer is added to a model's image encoder and the subsequent model is finetuned to make predictions using only 5% of the training data.

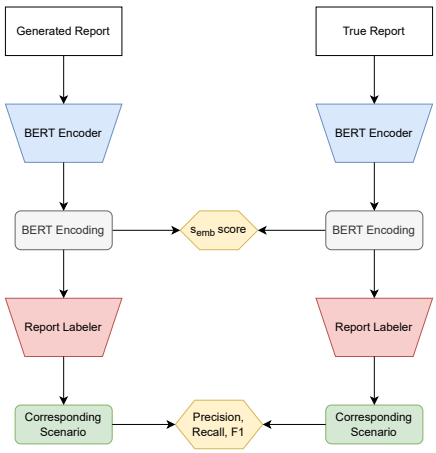

Figure 3: Evaluation metrics and how they are computed for the report generation task. We use the BERT embeddings directly to compute s_{emb}, and use the final classification of scenario type (shown here as the Report Labeler) to compute the Precision, Recall, and F1 scores.

Figure 3 details our evaluation metrics for the report generation task and how they are computed. Starting with a pretrained BERT model from HuggingFace [60, 58], we append a softmax classification layer and finetune the model by freezing the original BERT layers and training the output layer in order to classify reports based on the attack scenario they reference. We use the output classifications to compute the precision, recall, and f1 score metrics of results on the image-text task, chosen due to the imbalanced nature of our problem space. These metrics will allow for a better evaluation of models [67]. These metrics are computed globally by counting the total true positives, false negatives and false positives. Additionally, we use the BERT embeddings directly to compute the similarity embedding s_{emb} metric proposed by CXR-RePaiR [68], a medical report retrieval model. We will take the reports output by ReportIT as well as our baseline models, and the cosine similarity between the last hidden representations of the report when fed through the evaluation BERT model.

As we are focusing on the evaluation of various Natural Language Processing techniques, we use the Vision Image Transformer (ViT) [59] as our image encoder for ReportIT as well as all baseline models in order to remove the image encoding architecture as a potential confounding variable.

In order to obtain 95% confidence intervals, bootstrapping using 1000 samples of all metrics is applied for all quantitative metrics.

Experimental Details Across experiments, data was categorized into different sets at the user level within 24-hour windows via a stratified split, with 70% in the training set, 10% in the validation set, and 20% in the test set. Behavior image encodings are generated and data imbalance is handled via the same data augmentations described in the current best performing insider threat detection model

architecture [9]; we go over the full process in more detail in Appendix B. Each image is manually annotated with a text report based on the ground truth information found in the dataset. Augmented images use the same report as the original image.

All models start with pretrained checkpoints from Huggingface, with ReportIT models trained using the training set for 10 epochs while baselines are trained for 100. Number of epochs were determined based on training convergence. All models are trained using Adam [69], with lr = 0.001, betas = $(0.9, 0.999)$, eps = 10^{-8}, and weight_decay = 0. For the report generation task, our image captioning baselines employ a beam search decoding algorithm using 5 beams.

5 Results and Analysis

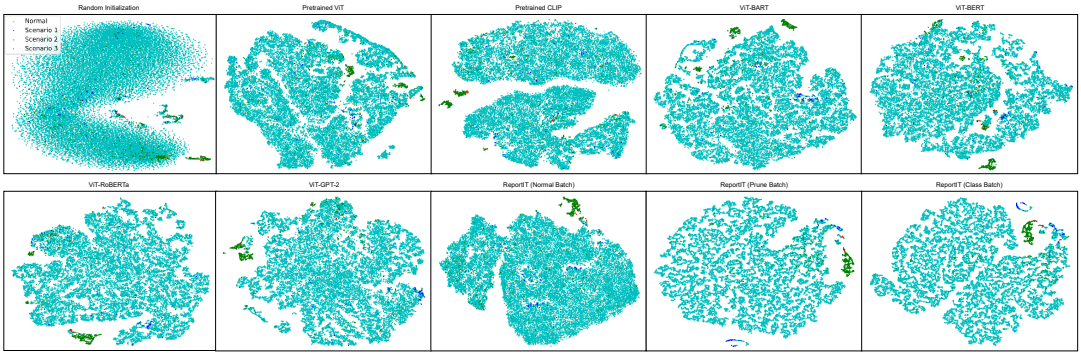

Figure 4: t-SNE visualizations for the various trained image encoders used. Normal class data has been randomly undersampled.

t-SNE Visualizations Figure 4 showcases the t-SNE visualization from the ViT image encoders as trained by the various models. We also include representations from a randomly initialized architecture, as well as pretrained ViT and pretrained CLIP model checkpoints from Huggingface. With the exception of ReportIT NormalBatch, all models that have been trained using text perform better than the encoding architectures that do not, showcasing greater separation of malicious behavior from benign data and having greater clustering altogether. The randomly initialized ViT visualization appears quite diffuse in appearance, while the pretrained ViT models have more distinct clusters, even beginning to separate more of the scenario 2 and 3 results from benign data. Both ReportIT PruneBatch and Report ClassBatch perform the best, showing better separation of scenario 1 data from benign data than all other models. ReportIT ClassBatch performs slightly better than ReportIT PruneBatch, having a larger number of scenario 1 and 2 embeddings clustered away from the benign embeddings.

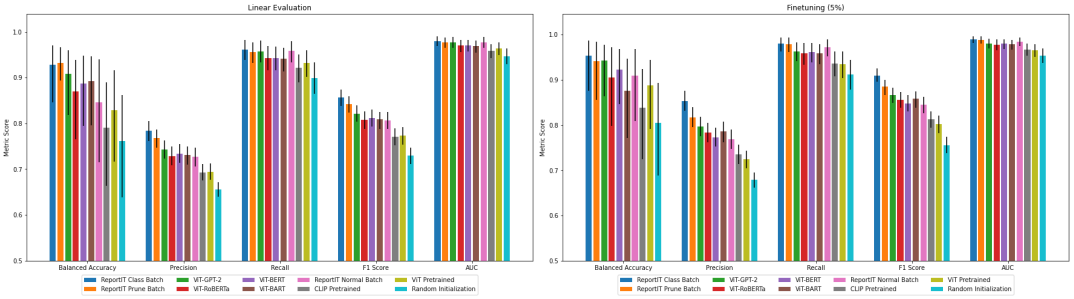

Figure 5: Results for linear evaluation (left) and finetuning (right) experiments.

Linear Evaluation and Finetuning Figure 5 and Tables 2 and 3 showcase the results for linear evaluation and finetuning. ReportIT ClassBatch outperforms all other models for both experiments, with the exception of balanced accuracy under linear evaluation where ReportIT PruneBatch performs better. The overall best performing image captioning model across both experiments is ViT-GPT2, even outperforming ReportIT PruneBatch in Recall and AUC under linear evaluation, and balanced accuracy

under finetuning. ReportIT NormalBatch performs well in Recall and AUC across both experiments, but performs worse than alternatives in the other metrics.

Here we again see the benefits of using text to train encoders, as the best performing model trained on text reports has a balanced accuracy that is 10% better and a precision that is 9% than the best performing model without this training on linear evaluation. A similar situation occurs after finetuning; the best performing model trained on text reports has a balanced accuracy that is 7% better and a precision that is 12% better than the best performing model without this training. These results are especially critical for insider threat detection as false positive in model detection mean an employee that did nothing wrong was classified as having malicious intent. Since this could lead to an employee losing his/her job, having a high precision is critical, and we see here we can achieve good precision using only a small subset of labelled data after pre-training using text.

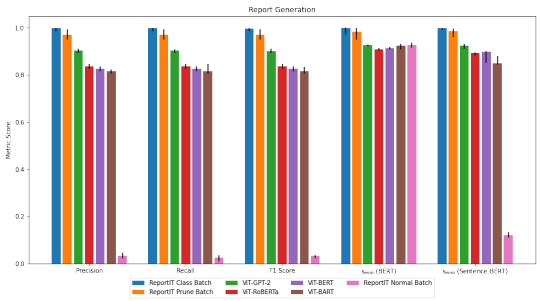

Figure 6: Results for the report generation task.

Report Generation Figure 6 and Table 4 showcase ReportIT in comparison to baseline models. As expected, using a normal batch configuration leads to subpar performance, whereas PruneBatch and ClassBatch allows ReportIT to better differentiate different image-text pairs, drastically improving report retrieval performance. Of the image captioning models, ViT-GPT2 outperforms the other models, which is to be expected due to GPT2 being specifically designed for the text generation task. ReportIT PruneBatch and ReportIT ClassBatch outperforms all baseline models, indicating that taking advantage of the relatively small space of potential insider threat reports leads to superior results. ReportIT ClassBatch slightly outperforms ReportIT PruneBatch, indicating that the class-based representation of text leads to superior image-text pairing.

The s_{emb} metric values computed via BERT are high for all model types evaluated, with ReportIT NormalBatch outperforming image captioning models despite performing significantly worse than them in all other metrics. However, this is due to the original s_{emb} metric proposed by by CXR-RePaiR [68] requiring a model designed for the sentence similarity task. The cosine similarity metric typically requires that all of the dimensions of a vector contribute equally, however this is not the case for the original BERT model which is a language model that was not designed for such tasks. Indeed, despite high sentence diversity among all possible reports, the lowest cosine similarity between BERT embeddings was 0.6874, and the mean cosine similarity was 0.9318. We thus elect to also report s_{emb} metrics derived from Sentence-BERT encodings for this analysis [70]. Using Sentence-BERT, we achieve much lower cosine similarities among reports; the lowest cosine similarity between Sentence-BERT embeddings was 0.0013, and the mean cosine similarity was 0.3651. Sentence-BERT s_{emb} metric values appear to more closely match the trends seen by Precision, Recall, and F1 regarding which models perform better than BERT s_{emb} metric values.

6 Conclusion

ReportIT takes the behavior image representations that have led to great detection accuracy and returns insider threat reports that explain why the given image is malicious if so. This improves the transparency of insider threat detection models, which can help reduce skepticism regarding their use and deployment in industry. Additionally, by pretraining ReportIT's image encoder using text reports, we increase the label efficiency of the image encoder, reducing the costs and effort related to creating the private datasets needed to deploy insider threat detection systems. We also propose two novel contrastive learning training methodologies. We showcase that ClassBatch in particular leads to impressive label efficiency and report generation results, indicating that this training methodology works well for multimodal contrastive learning when one modality has higher diversity than the other.

References

[1] Verizon. Data breach investigations report, 2018.

[2] Tarzey. The insider security threat manifesto: Beating the threat from within, 2019.

[3] Verizon Enterprise Solutions. Data breach investigations report (dbir), 2019.

[4] Ponemon Institute. 2020 cost of insider threats global report, 2020.

[5] Roberts. Massive tjx security breach reveals credit card data, 2007.

[6] Zetter. Verisign hit by hackers in 2010, 2012.

[7] Linn Foster Freedman. Adobe settles proposed class action data breach case with award of 1.18 million for plaintiffs attorneys, 2015.

[8] Nathan Mcalone. A hacker is reportedly selling the stolen emails and passwords of 117 million linkedin user, 2016.

[9] Sameer Khanna. Computer vision user entity behavior analytics. *arXiv preprint arXiv:2111.13176*, 2021.

[10] Steven Walker-Roberts, Mohammad Hammoudeh, and Ali Dehghantanha. A systematic review of the availability and efficacy of countermeasures to internal threats in healthcare critical infrastructure. *IEEE Access*, 6:25167–25177, 2018.

[11] Daniel S Berman, Anna L Buczak, Jeffrey S Chavis, and Cherita L Corbett. A survey of deep learning methods for cyber security. *Information*, 10(4):122, 2019.

[12] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[13] Gaurang Gavai, Kumar Sricharan, Dave Gunning, Rob Rolleston, John Hanley, and Mudita Singhal. Detecting insider threat from enterprise social and online activity data. In *Proceedings of the 7th ACM CCS international workshop on managing insider security threats*, pages 13–20, 2015.

[14] Liu Liu, Olivier De Vel, Chao Chen, Jun Zhang, and Yang Xiang. Anomaly-based insider threat detection using deep autoencoders. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 39–48. IEEE, 2018.

[15] David Noever. Classifier suites for insider threat detection. *arXiv preprint arXiv:1901.10948*, 2019.

[16] Mohammed Nasser Al-Mhiqani, Rabiah Ahmed, Z Zainal Abidin, and SN Isnin. An integrated imbalanced learning and deep neural network model for insider threat detection. *International Journal of Advanced Computer Science and Applications*, 12(1), 2021.

[17] H He, Y Bai, EA Garcia, and S ADASYN Li. adaptive synthetic sampling approach for imbalanced learning. ieee international joint conference on neural networks. In *2008 (IEEE World Congress On Computational Intelligence)*, 2008.

[18] Balaram Sharma, Prabhat Pokharel, and Basanta Joshi. User behavior analytics for anomaly detection using lstm autoencoder-insider threat detection. In *Proceedings of the 11th International Conference on Advances in Information Technology*, pages 1–9, 2020.

[19] Duc C Le, Nur Zincir-Heywood, and Malcolm Heywood. Training regime influences to semi-supervised learning for insider threat detection. In *2021 IEEE Security and Privacy Workshops (SPW)*, pages 13–18. IEEE, 2021.

[20] Fanzhi Meng, Fang Lou, Yunsheng Fu, and Zhihong Tian. Deep learning based attribute classification insider threat detection for data security. In *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pages 576–581. IEEE, 2018.

[21] Fangfang Yuan, Yanan Cao, Yanmin Shang, Yanbing Liu, Jianlong Tan, and Binxing Fang. Insider threat detection with deep neural network. In *International Conference on Computational Science*, pages 43–54. Springer, 2018.

[22] Lingli Lin, Shangping Zhong, Cunmin Jia, and Kaizhi Chen. Insider threat detection based on deep belief network feature representation. In *2017 International Conference on Green Informatics (ICGI)*, pages 54–59. IEEE, 2017.

[23] Jiange Zhang, Yue Chen, and Ankang Ju. Insider threat detection of adaptive optimization dbn for behavior logs. *Turkish Journal of Electrical Engineering & Computer Sciences*, 26(2):792–802, 2018.

[24] Pratik Chattopadhyay, Lipo Wang, and Yap-Peng Tan. Scenario-based insider threat detection from cyber activities. *IEEE Transactions on Computational Social Systems*, 5(3):660–675, 2018.

[25] Dorothy E Denning. An intrusion-detection model. *IEEE Transactions on software engineering*, (2):222–232, 1987.

[26] Carrie Gates and Carol Taylor. Challenging the anomaly detection paradigm: a provocative discussion. In *Proceedings of the 2006 workshop on New security paradigms*, pages 21–29, 2006.

[27] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[28] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936.

[29] Madhu Shashanka, Min-Yi Shen, and Jisheng Wang. User and entity behavior analytics for enterprise security. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1867–1874. IEEE, 2016.

[30] Larry Lunetta. The ciso's guide to machine learning and user and entity behavioral analytics, 2012.

[31] Jamie Graves. Insider threat protection. *ETSI Security Week*, 2019.

[32] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 635–644, 2011.

[33] Steffen Rendle. Factorization machines. In *2010 IEEE International conference on data mining*, pages 995–1000. IEEE, 2010.

[34] Immanuel Bayer. fastfm: A library for factorization machines. *The Journal of Machine Learning Research*, 17(1):6393–6397, 2016.

[35] Baoming Tang, Qiaona Hu, and Derek Lin. Reducing false positives of user-to-entity first-access alerts for user behavior analytics. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 804–811. IEEE, 2017.

[36] T Bussa, Avivah Litan, and T Phillips. Market guide for user and entity behavior analytics. *Gartner*, 2018.

[37] Alex Pinto. Secure because math: A deep-dive on machine learning-based monitoring. *Black Hat Briefings*, 25:1–11, 2014.

[38] Konrad Rieck. Computer security and machine learning: Worst enemies or best friends? In *2011 First SysSec Workshop*, pages 107–110. IEEE, 2011.

[39] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE symposium on security and privacy*, pages 305–316. IEEE, 2010.

[40] Minyoung Huh, Pulkit Agrawal, and Alexei A Efros. What makes imagenet good for transfer learning? *arXiv preprint arXiv:1608.08614*, 2016.

[41] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523, 2020.

[42] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. *Advances in neural information processing systems*, 32, 2019.

[43] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[44] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):4037–4058, 2020.

[45] Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. *Advances in neural information processing systems*, 33:8765–8775, 2020.

[46] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020.

[47] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823, 2020.

[48] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[49] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. Contrastive learning of medical visual representations from paired images and text. *arXiv preprint arXiv:2010.00747*, 2020.

[50] Minghao Li, Tengchao Lv, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. Trocr: Transformer-based optical character recognition with pre-trained models. *arXiv preprint arXiv:2109.10282*, 2021.

[51] Jun Chen, Han Guo, Kai Yi, Boyang Li, and Mohamed Elhoseiny. Visualgpt: Data-efficient adaptation of pretrained language models for image captioning. *arXiv preprint arXiv:2102.10407*, 2021.

[52] Viktar Atliha and Dmitrij Šešok. Pretrained word embeddings for image captioning. In *2021 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream)*, pages 1–4. IEEE, 2021.

[53] Vasiliki Kougia, John Pavlopoulos, and Ion Androutsopoulos. A survey on biomedical image captioning. *arXiv preprint arXiv:1905.13302*, 2019.

[54] Nikhil Patwari and Dinesh Naik. En-de-cap: An encoder decoder model for image captioning. In *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1192–1196. IEEE, 2021.

[55] MD Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CsUR)*, 51(6):1–36, 2019.

[56] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4651–4659, 2016.

[57] Jyoti Aneja, Aditya Deshpande, and Alexander G Schwing. Convolutional image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5561–5570, 2018.

[58] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[59] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[60] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[61] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[62] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[63] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[64] Joshua Glasser and Brian Lindauer. Bridging the gap: A pragmatic approach to generating insider threat data. In *2013 IEEE Security and Privacy Workshops*, pages 98–104. IEEE, 2013.

[65] Brian Lindauer, Joshua Glasser, Mitch Rosen, Kurt C Wallnau, and L ExactData. Generating test data for insider threat detectors. *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, 5(2):80–94, 2014.

[66] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[67] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C Prati, Bartosz Krawczyk, and Francisco Herrera. *Learning from imbalanced data sets*, volume 11. Springer, 2018.

[68] Mark Endo, Rayan Krishnan, Viswesh Krishna, Andrew Y Ng, and Pranav Rajpurkar. Retrieval-based chest x-ray report generation using a pre-trained contrastive language-image model. In *Machine Learning for Health*, pages 209–219. PMLR, 2021.

[69] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[70] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[71] Sameer Khanna. Conical classification for efficient one-class topic determination. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1662–1673, 2021.

[72] Kim T Gribbon and Donald G Bailey. A novel approach to real-time bilinear interpolation. In *Proceedings. DELTA 2004. Second IEEE International Workshop on Electronic Design, Test and Applications*, pages 126–131. IEEE, 2004.

[73] Zhaomin Chen, Chai Kiat Yeo, Bu Sung Lee, and Chiew Tong Lau. Autoencoder-based network anomaly detection. In *2018 Wireless Telecommunications Symposium (WTS)*, pages 1–5. IEEE, 2018.

[74] Andrew Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19, 2011.

[75] Kymie MC Tan, Kevin S Killourhy, and Roy A Maxion. Undermining an anomaly-based intrusion detection system using common exploits. In *International Workshop on Recent Advances in Intrusion Detection*, pages 54–73. Springer, 2002.

[76] Iffat A Gheyas and Ali E Abdallah. Detection and prediction of insider threats to cyber security: a systematic literature review and meta-analysis. *Big Data Analytics*, 1(1):1–29, 2016.

[77] Amos Azaria, Ariella Richardson, Sarit Kraus, and VS Subrahmanian. Behavioral analysis of insider threat: A survey and bootstrapped prediction in imbalanced data. *IEEE Transactions on Computational Social Systems*, 1(2):135–155, 2014.

[78] Naghmeh Moradpoor Sheykhkanloo and Adam Hall. Insider threat detection using supervised machine learning algorithms on an extremely imbalanced dataset. *International Journal of Cyber Warfare and Terrorism (IJCWT)*, 10(2):1–26, 2020.

[79] RG Gayathri, Atul Sajjanhar, and Yong Xiang. Image-based feature representation for insider threat classification. *Applied Sciences*, 10(14):4945, 2020.

[80] Shuhan Yuan and Xintao Wu. Deep learning for insider threat detection: Review, challenges and opportunities. *Computers & Security*, page 102221, 2021.

A Related Works (continued)

In this section, we expound on the methodologies of algorithms proposed by academia cited in the Related Works section of the main manuscript.

Gavai et al. [13] proposed an Isolation Forest-based unsupervised approach for detecting insider threats using network logs. They aggregate features that contribute most to the isolation of a data sample within the tree to better ascertain why a user was tagged as anomalous.

Liu et al. [14] proposed an ensemble of deep autoencoders to unravel the non-linear relationships in log data. A model is built from each autoencoder based on the extracted features from each log file. Finally, the outputs are aggregated into a single model used to detect malicious insider activities. Unfortunately this procedure has its limitations: returning subpar results for datasets from alternative sources, the frequency based feature extraction methodology does not always provide the expected outcome, and the one-hour interval considered for user behavior study does not provide enough resolution to identify usage patterns.

Noever et al. [15] tried a variety of different learning algorithm families, concluding that Random Forests with Randomization and Boosted Logistic Regression provided the best results after extracting risk factors from data to create their feature vectors. While their results indicate that Boosted Logistic Regression outperformed the former algorithm, Noever et al. advocate for the usage of Random Forests in insider threat detection systems as they offer a deep but human-readable set of detection rules.

Noting that the vast majority of implementations suggested in recent publications suffer from a very low accuracy of the minority class due to extreme class imbalance, Al-Mhiqani et al. [16] proposed an intuitive way to tackle this issue. They combine adaptive synthetic sampling (AD) [17] with a deep neural network (DNN) architecture to develop AD-DNN, an integrated model that improves the overall detection performance of insider threats.

Sharma et al. [18] used a two-step process to detection via their Long Short-Term Memory Autoencoder (LSTM-Autoencoder). First, it calculates the reconstruction error using the autoencoder fit on normal data, and then utilizes a threshold based detection scheme to identify outliers. The identified outliers are then classified as malicious behavior.

Le et al. [19] assessed various semi-supervised learning algorithms in conjunction with different labeled data availability conditions. These conditions are designed to emulate real-world situations representing the availability of various scenarios of ground truth.

Meng et al. [20] combined Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) and Kernel Principal Component Analysis (PCA) for analysis of insider behavior. They compared well against popular algorithms such as Support Vector Machines (SVM) and Isolation Forests, however it is important to note that their approach was not compared with deep learning models.

Yuan et al. [21] identified that a user action sequence has a temporal dependency. They feed these sequences to a Long Short-Term Memory (LSTM) network, which extracts user behavior features and predicts the next user action. Various hidden states of the LSTM are then used to develop a fixed size representation passed to a Convolutional Neural Network (CNN) for detection purposes.

It is beneficial to identify user behavior patterns within multi-domain scenarios. However, incorporating multi-domain irrelevant features may hide the existence of anomalies within our data. Thus, Lin et al. [22] formulated a hybrid method using Deep Belief Networks (DBN) for unsupervised feature reconstruction, and One Class SVM (OCSVM) for insider threat detection. The usage of the DBN provides a substantial performance uplift when compared to using OCSVM by itself, indicating a promising direction for insider threat detection.

Lin et al. are not the only team that proposed using this network family; Zhang et al. [23] also focused on using a DBN network, albeit within a supervised regime. First, feature learning is executed by having each layer trained using the unsupervised learning method and the training results are adopted as the input of the next layer. Finally, the entire network is fine-tuned by using supervised training. The final output is determined after being fed through a softmax output layer.

Chattopadhyay et al. [24] proposed an insider threat detection approach based on classification of time-series user activities. A cost-sensitive technique for data adjustment was used to randomly undersample the instances belonging to the minority class. A deep autoencoder with two layers and a threshold parameter was used for classification.

Khanna [9] noted the that traditional image encodings used by alternative methodologies did not satisfy the weight sharing assumption required by the convolutional neural networks such methods relied on, leading to brittle networks that performed poorly on unforseen attacks. Using the Natural Language Processing model Conical Classification [71] to extract various features from log files, Khanna proposed transforming the insider threat problem into a color detection problem using a novel procedure to encode human behavior, leading to state of the art performance when used in tandem with a dual-input classifier model.

B Generating Behavior Images

Representing Text as Greyscale Images Unlike traditional image encoding implementations that utilizes interpolation [72] to facilitate this process, the state of the art approach Computer Vision User Entity Behavior Analytics (CVUEBA) uses Sparse AutoEncoders.

SAEs in anomaly detection are typically trained on normal data only. The expectation is that the reconstruction error will be noticeably higher on anomalies than it will on normal data, as anomalies will be encoded differently and thus will be distributed away from normal data. A threshold parameter is then used to separate vectors into normal and anomalous classes [73]. CVUEBA instead uses the trained SAE hidden layer to automatically learn better feature representations from the given data [74].

To transform the SAE output values into a range of 0-255 suitable for images, CVUEBA first applies Min-Max Scaling and proceed to multiply all values by 255. Both of these actions are done within the model's layers itself; testing has shown that performing the scaling in this manner improves speed, as well as storage and memory requirements for training. The SAE has a hidden dimension of size 1024; once output encodings are scaled, they are reshaped into 32x32x1 images. Figure 7a details the full process used to convert text based log data into greyscale images.

Context-Channel Representations While we could feed these greyscale images directly into a model for attack identification, it would be beneficial to have contextual information regarding the typical behavior of a user. This would enable us to identify the sudden behavioral changes indicative of an attack and mitigates the concerns regarding insider threat detection systems brought up by Tan et al. [75]; by comparing a user's behavior to their own previous behavior rather than performing the comparison at the activity level via a detection regime, we mitigate the potential of attackers taking advantage of detection loopholes and acting undetected.

CVUEBA provides this information by passing in behavior encodings for the previous two days in addition to the current day we are evaluating. As shown in Figure 7b, we append the encoding for each day as a different channel, leaving us with a color image for evaluation purposes.

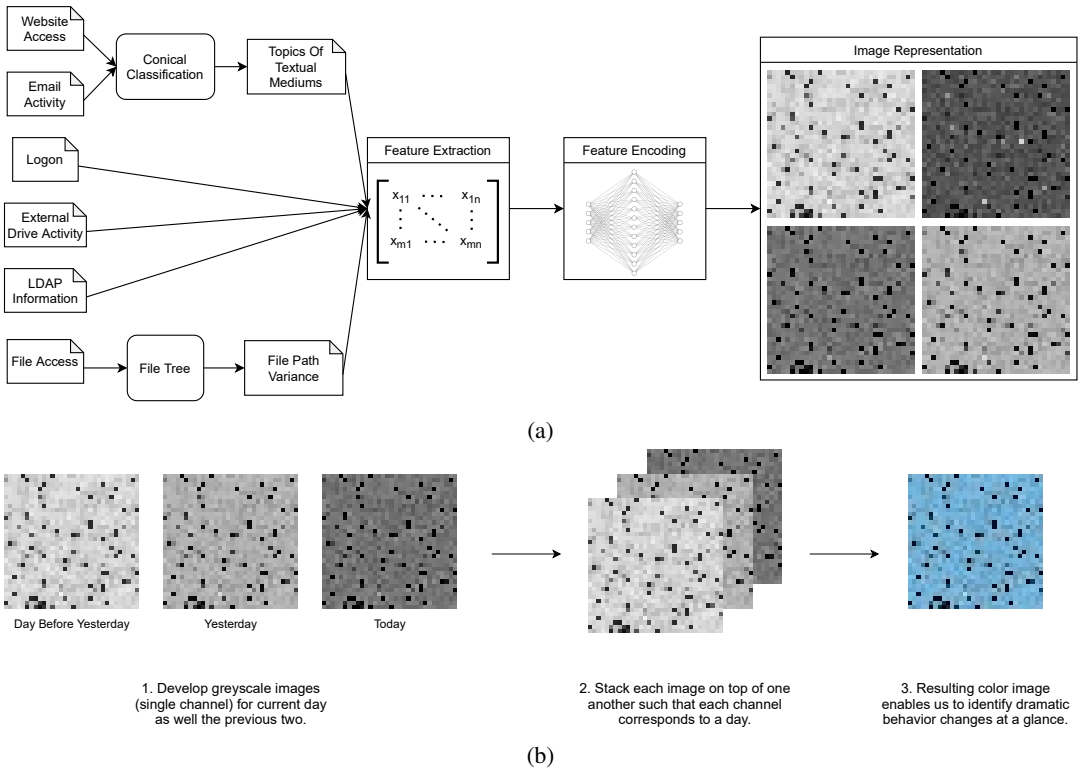

Figure 7: Generating Color Image Encodings From Start to Finish. (a) Process flow from log files to greyscale images. (b) Representing context behavior encodings as channels.

If a user is not acting malicious, there is little to no variation in behavior features from a day to day basis; this leaves images representing benign behavior fairly grey in appearance. On the other hand, malicious actions will indeed have changes in observed behavior, which will lead to colorful images.

This phenomenon is due to how RGB images work. The RGB scale is calibrated so that when a given pixel's red-green-blue numbers are equal, the pixel is represented as a shade of gray where larger red-green-blue numbers lead to lighter shades of grey. The CVUEBA SAE model is trained such that the red-green-blue numbers across all dimensions will be fairly similar to one another if a user's behavior is benign. Malicious behavior on the other hand will lead to drastic differences in the red-green-blue numbers as the current information varies significantly from the contextual information which leads to the image having a more colorful representation.

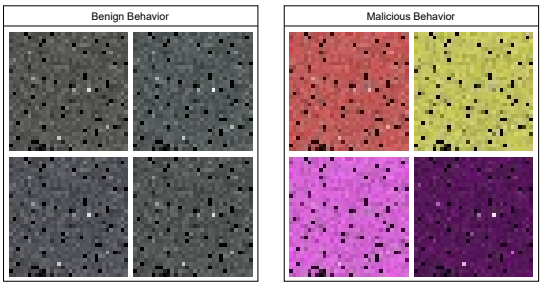

Figure 8: Benign vs Malicious Images.

As we can see in Figure 8, this allows malicious image representations to be identifiable, even by those who are not security experts.

Context Changing Data Augmentation As the vast majority of employees within a company are good-natured and do not have malicious intent [76, 77, 78, 16, 79], insider threat detection is a highly

imbalanced problem space. This poses problems for potential classification models, as during training models will spend most of their time on the dominant class and will fail to learn enough from the minority classes; decision boundaries either become too complex and we lose the ability to generalize to unseen data, or minority sub-concepts are ignored altogether due to not providing enough discriminative information to classifiers [67]. The data imbalance problem is one of the most important unsolved challenges for current insider threat detection systems [80].

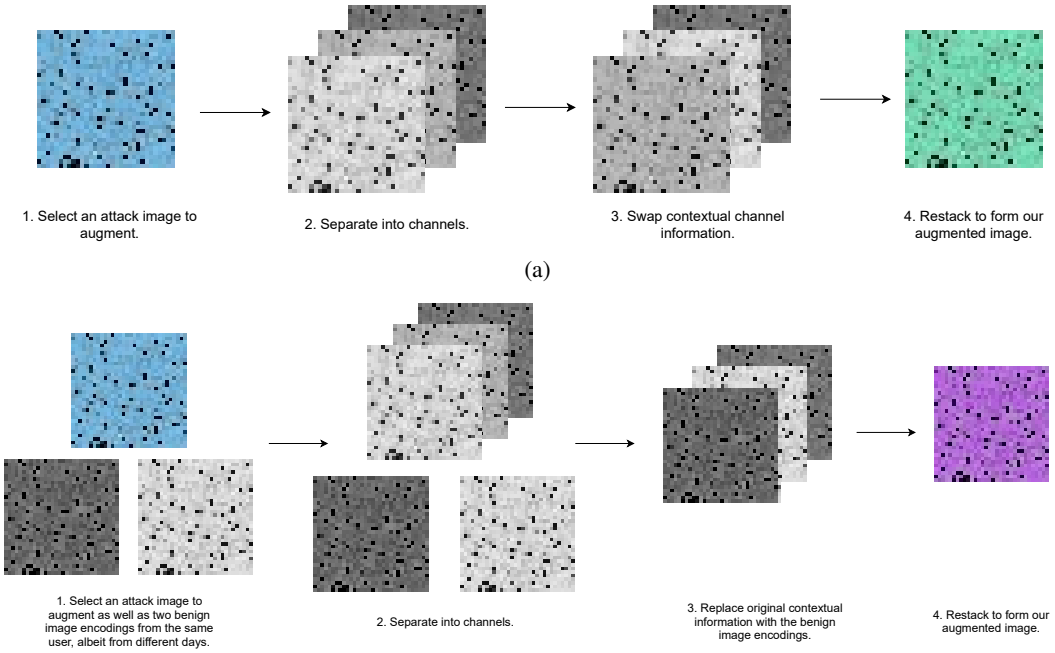

Figure 9: Context Changing Data Augmentations. (a) Channel swapping augmentation. (b) Channel replacement augmentation.

Note that two channels of each image consist solely of information that allows us to identify if the current day's behavior is malicious. Thus, we are able to swap out information held in these channels and still possess a valid representation. CVUEBA augments our training set by swapping the positions of the contextual channels; the channel corresponding to yesterday becomes the day before yesterday, and the channel corresponding to the day before yesterday becomes yesterday. Additionally, CVUEBA also randomly select days of benign behavior for the given user and use these to replace our contextual information. Both forms of data augmentation performed as well as examples of final results are illustrated in Figure 9.

C Results Tables

Tables providing numerical values for the results found in the results figures of the main manuscript can be found below. The best performing value for each given metric is bolded.

Table 2: Baselines vs ReportIT Linear Evaluation. Values in brackets correspond to 95% confidence interval.

Model	Balanced Accuracy	Precision	Recall	F1 Score	AUC
ReportIT Class Batch	0.9282 [0.8458, 0.9703]	**0.7842 [0.7616, 0.8054]**	**0.9612 [0.9389, 0.9815]**	**0.8565 [0.8375, 0.8740]**	**0.9796 [0.9684, 0.9896]**
ReportIT Prune Batch	**0.9324 [0.8932, 0.9665]**	0.7673 [0.7463, 0.7869]	0.9563 [0.9317, 0.9765]	0.8422 [0.8238, 0.8588]	0.9770 [0.9645, 0.9870]
ViT-GPT-2	0.9080 [0.8182, 0.9592]	0.7428 [0.7235, 0.7630]	0.9575 [0.9331, 0.9803]	0.8213 [0.8033, 0.8394]	0.9773 [0.9651, 0.9887]
ViT-RoBERTa	0.8694 [0.7657, 0.9388]	0.7289 [0.7088, 0.7495]	0.9429 [0.9158, 0.9684]	0.8072 [0.7879, 0.8262]	0.9698 [0.9562, 0.9825]
ViT-BERT	0.8865 [0.7942, 0.9481]	0.7337 [0.7137, 0.7541]	0.9429 [0.9159, 0.9676]	0.8116 [0.7928, 0.8303]	0.9699 [0.9565, 0.9824]
ViT-BART	0.8918 [0.7954, 0.9469]	0.7304 [0.7096, 0.7491]	0.9418 [0.9132, 0.9655]	0.8085 [0.7872, 0.8253]	0.9693 [0.9549, 0.9811]
ReportIT Normal Batch	0.8460 [0.7151, 0.9404]	0.7264 [0.7064, 0.7462]	0.9584 [0.9326, 0.9800]	0.8065 [0.7881, 0.8253]	0.9775 [0.9647, 0.9884]
CLIP Pretrained	0.7900 [0.6639, 0.8894]	0.6932 [0.6753, 0.7111]	0.9219 [0.8896, 0.9507]	0.7711 [0.7517, 0.7897]	0.9588 [0.9426, 0.9731]
ViT Pretrained	0.8288 [0.7167, 0.9164]	0.6940 [0.6761, 0.7127]	0.9316 [0.9010, 0.9592]	0.7728 [0.7538, 0.7914]	0.9637 [0.9484, 0.9774]
Random Initialization	0.7616 [0.6390, 0.8614]	0.6552 [0.6391, 0.6708]	0.8992 [0.8644, 0.9327]	0.7293 [0.7106, 0.7470]	0.9467 [0.9292, 0.9633]

Table 3: Baselines vs ReportIT Finetuning. Values in brackets correspond to 95% confidence interval.

Model	Balanced Accuracy	Precision	Recall	F1 Score	AUC
ReportIT Class Batch	**0.9533 [0.8752, 0.9871]**	**0.8527 [0.8320, 0.8759]**	**0.9800 [0.9631, 0.9934]**	**0.9098 [0.8954, 0.9251]**	**0.9894 [0.9811, 0.9961]**
ReportIT Prune Batch	0.9412 [0.8557, 0.9838]	0.8173 [0.7954, 0.8390]	0.9793 [0.9610, 0.9934]	0.8845 [0.8671, 0.9002]	0.9888 [0.9796, 0.9959]
ViT-GPT-2	0.9430 [0.8643, 0.9780]	0.7971 [0.7755, 0.8185]	0.9627 [0.9414, 0.9826]	0.8668 [0.8492, 0.8829]	0.9804 [0.9698, 0.9904]
ViT-RoBERTa	0.9048 [0.7977, 0.9716]	0.7831 [0.7622, 0.8044]	0.9584 [0.9340, 0.9815]	0.8553 [0.8377, 0.8729]	0.9781 [0.9659, 0.9897]
ViT-BERT	0.9225 [0.8461, 0.9685]	0.7733 [0.7525, 0.7944]	0.9618 [0.9384, 0.9821]	0.8478 [0.8298, 0.8658]	0.9798 [0.9680, 0.9899]
ViT-BART	0.8754 [0.7711, 0.9474]	0.7866 [0.7623, 0.8074]	0.9591 [0.9349, 0.9795]	0.8582 [0.8379, 0.8748]	0.9785 [0.9664, 0.9887]
ReportIT Normal Batch	0.9086 [0.8090, 0.9679]	0.7681 [0.7467, 0.7896]	0.9724 [0.9515, 0.9897]	0.8449 [0.8265, 0.8630]	0.9849 [0.9745, 0.9936]
CLIP Pretrained	0.8377 [0.7242, 0.9246]	0.7355 [0.7143, 0.7563]	0.9358 [0.9085, 0.9635]	0.8124 [0.7935, 0.8307]	0.9663 [0.9528, 0.9802]
ViT Pretrained	0.8882 [0.7919, 0.9438]	0.7243 [0.7040, 0.7436]	0.9348 [0.9040, 0.9623]	0.8022 [0.7822, 0.8206]	0.9657 [0.9503, 0.9794]
Random Initialization	0.8042 [0.6886, 0.8926]	0.6788 [0.6614, 0.6954]	0.9122 [0.8789, 0.9446]	0.7556 [0.7374, 0.7744]	0.9537 [0.9371, 0.9700]

Table 4: Baselines vs ReportIT Report Generation. Values in brackets correspond to 95% confidence interval.

Model	Precision	Recall	F1 Score	s_{emb} (BERT)	s_{emb} (Sentence-BERT)
ReportIT Class Batch	**0.9987 [0.9854, 0.9988]**	**0.9986 [0.9862, 0.9988]**	**0.9960 [0.9847, 0.9988]**	**0.9994 [0.9718, 0.9999]**	**0.9993 [0.9916, 0.9994]**
ReportIT Prune Batch	0.9712 [0.9500, 0.9920]	0.9690 [0.9500, 0.9920]	0.9687 [0.9500, 0.9920]	0.9824 [0.9500, 0.9970]	0.9844 [0.9608, 0.9949]
ViT-GPT-2	0.9019 [0.8934, 0.9100]	0.9018 [0.8932, 0.9092]	0.9016 [0.8928, 0.9096]	0.9259 [0.9216, 0.9298]	0.9231 [0.9101, 0.9309]
ViT-RoBERTa	0.8363 [0.8256, 0.8464]	0.8367 [0.8262, 0.8462]	0.8362 [0.8254, 0.8462]	0.9082 [0.9029, 0.9131]	0.8928 [0.8832, 0.8942]
ViT-BERT	0.8266 [0.8160, 0.8366]	0.8266 [0.8160, 0.8366]	0.8263 [0.8164, 0.8364]	0.9133 [0.9081, 0.9183]	0.8974 [0.8518, 0.9000]
ViT-BART	0.8156 [0.8060, 0.8234]	0.8156 [0.8060, 0.8465]	0.8158 [0.8064, 0.8343]	0.9233 [0.9081, 0.9302]	0.8489 [0.8431, 0.8807]
ReportIT Normal Batch	0.0313 [0.0220, 0.0460]	0.0254 [0.0105, 0.0347]	0.0323 [0.0213, 0.0358]	0.9254 [0.9356, 0.9155]	0.1192 [0.1108, 0.1327]