A Novel Criterion of Reconstruction-based Anomaly Detection for Sparse-binary Data

Heng Qiao
Electrical and Computer Engineering
University of Florida
Gainesville, USA
hengqiao@ufl.edu

Daniela Oliveira
Electrical and Computer Engineering
University of Florida
Gainesville, USA
daniela@ece.ufl.edu

Dapeng Wu
Electrical and Computer Engineering
University of Florida
Gainesville, USA
dpwu@ufl.edu

***Abstract*—Computer usage behaviour information can be used by anomaly detection algorithms to identify the current user of the computer system for security reasons. However, the data collected in this setup can be binary and very sparse, resulting in poor performance for some widely used anomaly detection methods. In this study, we propose a novel reconstruction criterion inspired by the F_1 score and the cross-entropy loss, that tackles the class imbalance problem introduced by binary and sparse data distribution with effectively merging reconstruction criterion calculated from vector elements of both positive and negative classes. Our experiments show that the proposed criterion can effectively improve the performance of reconstruction based anomaly detection methods, including both the PCA and the autoencoder.**

I. Introduction

Anomaly detection is a widely studied problem with a wide range of applications, one of which is anomaly detection in the computer system. Specifically, a computer system, on many occasions, is supposed to be used by one user and one user only for security purposes. The behaviour information can be used with anomaly detection algorithms in building systems to detect the identity of the current user. They can be realized by building a one-class classifier trained with the user behaviour information, which is collected through checking whether any entry in a pre-determined list of concerned applications and IP addresses is activated. The resulting data matrices used to train the one-class classifier model can thus be very sparse, leading to negative effects on the classification performance.

The one class classification refers to the problem that, in the training process, only the data of one class is provided, while at the test phase, the model should determine whether a testing sample belongs to the original class or not. From here on, we use the term one-class classification and anomaly detection interchangeably.

The one class classification is an active research area with many classic approaches proposed. A straight forward idea is to determine the class of a data sample through its distance to other data points [3] [1]. These approaches suffer from the curse of dimensionality since the distance measure generally will not work well on high dimensional data. Another class of widely used methods is the domain-based approach, where the trained model is essentially trying to find the hypersphere that bounds the normal training data [11] or a hyperplain that separates the normal training data from original point [10]. Being very successful and the most widely used methods for one-class classification, these methods are not specially designed to deal with sequential data. Another class of methods is the reconstruction based one-class classification, where the model is trained to map the input sample into a lower-dimensional space through minimizing the reconstruction loss. Since the model is only trained with samples from one class, when fed with samples from other classes, the reconstructive error of the output would be higher, and thus can be used as a criterion to determine whether it is from the original class. As a widely used dimension reduction method, the principal component analysis (PCA) can naturally be used as a reconstruction error based method, but it also does not fit well with sequential data in large size. The autoencoder [6], on the other hand, could combine with popular deep learning models such as GRU [2] or LSTM [4], and explore the potential hidden temporal information in the data [7] [12] [9]. These approaches, however, do not consider the situation when the input data is sparse. Contrary to widely used the cross-entropy loss or the MSE loss, the F_1 score has a natural potential of combining imbalanced class, which is induced by data sparsity, more effectively.

In this work, our main contribution is proposing a new reconstruction criterion inspired by the F_1 score and cross-entropy loss, which is able to improve the performance of the reconstruction-error-based one-class classification method when the input data is binary and sparse. Our formulation of the proposed criterion also leaves space for potential further optimization.

The remaining of this paper is organized as follows: Section II introduces the formulation of our data set and explains the sparse problem. In Section III, we describe the reconstruction-error-based method and our proposed reconstruction criterion in detail. The experiment results are shown in Section IV. Finally, Section V concludes the paper.

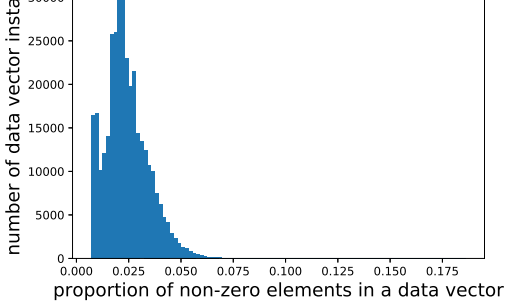


Fig. 1. Histogram of different levels of sparsity in our data set.

II. Data formulation

A. Data Collection and Sparsity

In this problem, the usage behaviour information data were collected from 17 volunteer subjects, and for each subject a software was used to continuously monitor what applications are activated and what IP addresses are visited in a pre-selected list of applications and IP addresses. The data of each time point are organized as one vector $v \in \mathbb{R}^m$ of 0s and 1s, with 1s indicating that the corresponding applications/IP addresses are activated or visited at the given time and 0s otherwise. In this task, about three weeks of user data were collected for 17 users. There are 554 interested applications and websites in the list and 10 columns are used to record time, thus $m = 554 + 10 = 564$. Because the list of interested application and IP addresses are pre-selected, the data matrices are very sparse. We calculated the percentage of non-zero elements in the data vectors on each time step, as presented in Figure 1. We can find that for most of the data vectors (one row in the matrices), only around 2% of the columns are non-zero. The average non-zero column percentage is 1.8%.

1) Sliding Window: Considering to exploit the possible temporal information in our data set, we used the sliding window technique, where data points of consecutive w time steps were considered as one and are assigned with one output label, being anomalous or not. For the model such as GRU-autoencoder that has the inherent structural ability to deal with sequential data, we fed data sequence into the model one step at a time, and for other models that are designed to take single vector input, we concatenate the data sequence into one vector and fed into them. When the size of the sliding window increase, the classification accuracy goes up, since more information was taken into account, but at the same time, the sensitivity of the system would drop since it takes more time to make one decision. For balancing two factors, here we chose window size to be 5.

III. Method

A. Background

Reconstruction-based methods for anomaly detection are usually comprised of two phases: the encoding and the decoding phase. When an input sample is fed to the model, it is first compressed into a lower dimension space. The low dimension representation of the input sample is then decoded back to the original space, as a reconstruction of the original input. A criterion function, usually known as reconstruction error, is used as a measure of reconstruction quality. After trained with only normal samples, the model would be forced to learn the underlying distributional information of the normal data set as it has to reconstruct the input from a lower dimension representation. As an anomaly detection system, when an anomalous test sample is given to the model, the reconstruction error would be larger than that of a normal sample, since the anomalous sample comes from a distribution that model has never seen before. We can then use the reconstruction error of test samples, combined with a threshold, as an indicator of whether the test sample is anomalous.

The model used in the encoding and decoding phases can take various forms. In our experiment, we used two different settings: GRU-autoencoder and PCA.

The GRU [2], similar to LSTM [4], is a variation of the classic Recurrent Neural Network (RNN), which is known to be able to process temporal information of longer sequences. In the GRU-autoencoder model, both encoder and decoder are the one-layer GRU network. The hidden state on the last time step of the encoder is fed into the decoder as the initial hidden state, and the decoder input is set to 0.

1) Reconstruction error: The reconstruction error of such model is often chosen to be the mean square error (MSE), or the ℓ_2 loss, that is, the error for the i-th input-reconstruction part can be defined as:

$$\ell_{mse}^{(i)} = \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\|_2, \tag{1}$$

where $\boldsymbol{y}_i$ is the i-th input test sequence, and $\hat{\boldsymbol{y}}_i$ is the corresponding reconstructed sequence. Considering that the input sequences are binary, we can also use the cross-entropy loss, which is defined as:

$$\ell_{ce}^{(i)} = -\sum_{k=1}^{p} y_{ik} \log(\hat{y}_{ik}) + (1 - y_{ik}) \log(1 - \hat{y}_{ik}), \tag{2}$$

where $\boldsymbol{y}_{ik}$ is the k-th element in the i-th input sample $\boldsymbol{y}_i$. Note that when computing the reconstruction error, the data sequence is flattened into a vector, so that $p = w \times m$. Recall that m is the input data dimension in each time step, and w is the window size.

2) Class imbalance: When the cross-entropy loss, which is often used as a criterion for the classification problem, is applied element-wise as reconstruction error, one interpretation is that we treat each element in the sequence as a binary

TABLE I

AUC ROC of different model/criterion combinations. The number was calculated by averaging th t.

	ℓ_{ce}^{+} (GRU-autoencoder)	ℓ_{ce}^{-} (GRU-autoencoder)
average AUC	0.8377	0.8111

classification input/output. This interpretation naturally leads to the discovering of the class-imbalance problem. Let

$$\begin{aligned}\ell_{ce}^{(i)+} &= -\sum_{k\in\mathbb{A}_i} y_{ik}\log(\hat{y}_{ik}) + (1-y_{ik})\log(1-\hat{y}_{ik}) \\ \ell_{ce}^{(i)-} &= -\sum_{k\in\mathbb{A}_i^c} y_{ik}\log(\hat{y}_{ik}) + (1-y_{ik})\log(1-\hat{y}_{ik}),\end{aligned} \tag{3}$$

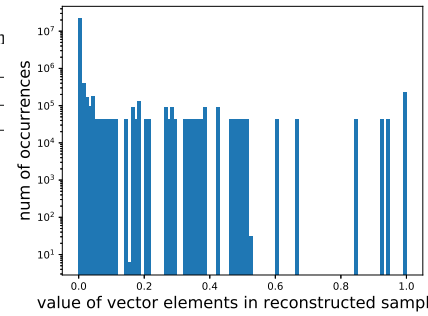
where set $\mathbb{A}_i \triangleq \{k|k \in [1,2,...,p], y_{ik} = 1\}$, then $\mathbb{A}_i^c \triangleq \{k|k \in [1,2,...,p], y_{ik} = 0\}$, accordingly. Then we have $\ell_{ce}^{(i)} = \ell_{ce}^{(i)+} + \ell_{ce}^{(i)-}$. Due to the heavy class imbalance introduced by the sparsity property in our data, $\ell_{ce}^{(i)-}$ is inferior on serving as criterion compared with $\ell_{ce}^{(i)+}$, as shown in Table I. Given that regardless of the input sample being anomalous or not, the percentage of positive elements in the reconstructed vectors would be small, thus the difference in misclassification between normal and anomalous samples represented in $\ell_{ce}^{(i)-}$ would be flooded by the correctly classified elements, which brings more noise and less useful information to $\ell_{ce}^{(i)-}$. Note that Table I also shows that the difference of ℓ_{ce}^{+} and ℓ_{ce}^{-} is more significant in the PCA model than in the GRU-autoencoder. This can be explained by the reconstruction distribution of each model, as shown in Figure 2, in which we can see that when smaller than the selected threshold of 0.5, the output distribution of GRU-autoencoder is flatter than that of PCA, which drops drastically as the data value approaches the threshold. This means that there is more information packed in the true-negatives of the GRU-autoencoder, since there is potentially more bad-correct classification (the ones close to the threshold). Thus, the performance of l_{ce}^{-} is in turn closer to l_{ce}^{+} compared with that of PCA.

The cross-entropy loss $\ell_{ce}^{(i)} = \ell_{ce}^{(i)+} + \ell_{ce}^{(i)-}$ is in turn not optimal because the inferior criterion, $\ell_{ce}^{(i)-}$, actually affects the final score more, since its mean and variance are larger than that of $\ell_{ce}^{(i)+}$. Thus, we need a better way of combining the residual criterion computed on positive and negative input elements than simply adding them together.

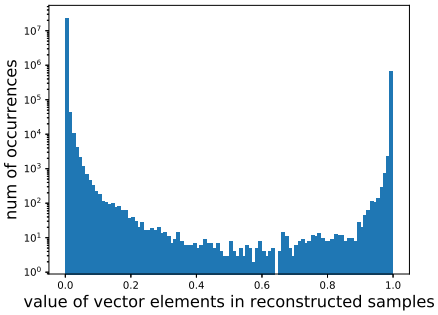
3) Weighted Sum: It is natural to think of using a weighted sum to tackle problems introduced by class-imbalance, that is,

$$\ell_{ce}^{(i)} = \beta\ell_{ce}^{(i)+} + \ell_{ce}^{(i)-}, \tag{4}$$

where β is the weight parameter, and should be set to a value larger than 1, since we are trying to emphasis $\ell_{ce}^{(i)+}$ more. However, in this case, the weighted sum does not work. Our experiments show that using weighted loss would in fact has detrimental effects on the classification performance.



(a) Reco



(b) Reconstruction distribution of PCA

Fig. 2. Reconstruction distribution of two models when trained with subject 10 (The distribution of other subjects have a very similar pattern). The histogram was drawn regarding the value of each element in the reconstructed data sequence vectors.

To understand the fail of using weighted sum, we can re-formulate cross-entropy loss as

$$\begin{aligned}\ell_{ce}^{(i)} &= \sum_{k=1}^{p}\varphi(r_{ik}) \\ \varphi(x) &= -\log(1-x),\end{aligned} \tag{5}$$

where $r_{ik} \triangleq |y_{ik} - \hat{y}_{ik}|$ is the residual of the k-th element in sample i. This way we unify the cross entropy loss with MSE, since the later can be written as

$$\begin{aligned}\ell_{ce}^{(i)} &= \sum_{k=1}^{p}\varphi(r_{ik}) \\ \varphi(x) &= x^2\end{aligned} \tag{6}$$

The goal of this representation of cross-entropy loss is to get rid of the information introduced in y_{ik} and see each element equally through residual r_{ik}. By re-formulating the cross-entropy loss, we find that using weighted loss is essentially putting more weight on a subset of residuals, resulting

in distortion in measuring the quality of reconstruction, and thus making it a worse criterion than not having weighted sum. The reformulation is also useful in building our proposed reconstruction criterion, as described in the next section.

B. $F_{1,log}$ score

Here we propose a new criterion that better combines criterion information from both positive and negative input elements inspired by cross-entropy loss and the F_1 score. First, we define the element-wise F_1 score for i-th input-reconstruction pair as

$$\begin{aligned} F^{(i)}_{1,elem} &= 2 \cdot \frac{precision_i \cdot recall_i}{precision_i + recall_i} \\ precision_i &= \frac{TP_i}{TP_i + FP_i} \\ recall_i &= \frac{TP_i}{TP_i + FN_i}, \end{aligned} \tag{7}$$

where TP_i, FP_i ,and FN_i are element-wise true-positive, false-positive and false-negative ,respectively, that is, we go through $w \times m$ elements in the flattened i-th input/reconstruction sequence and count how many got correctly or mistakenly classified. Note that since the reconstruction of an input sample is not Bernoulli-distributed, a threshold th is needed for counting.

$$\begin{aligned} TP_i &= \sum_{k=1}^{p} \mathbb{1}_{y_{ik}=1, \hat{y}_{ik} > th}(k) \\ FP_i &= \sum_{k=1}^{p} \mathbb{1}_{y_{ik}=0, \hat{y}_{ik} > th}(k) \\ FN_i &= \sum_{k=1}^{p} \mathbb{1}_{y_{ik}=1, \hat{y}_{ik} \leq th}(k), \end{aligned} \tag{8}$$

Commonly, the threshold is chosen to be 0.5. Note that when we use $F^{(i)}_{1,elem}$ as reconstruction criterion, it is no longer a loss, since better reconstruction yields higher $F^{(i)}_{1,elem}$ score. An obvious problem with $F^{(i)}_{1,elem}$ is that, similar to traditional F_1 score, it ignores the difference of reconstruction value except being larger or smaller than the threshold, e.g., an element in the reconstructed vector with a value of 0.1 or 0.3 would result in the same contribution on $F^{(i)}_{1,elem}$ value, while they, in fact, are of different reconstruction quality. To reduce the crudeness of $F^{(i)}_{1,elem}$, we can sum up the value of reconstruction directly after applying the threshold, resulting in a "softened" F_1 score $F^{(i)}_{1,soft}$, whose definition is identical to that of $F^{(i)}_{1,elem}$ except for the method to calculate true-positive, false-positive and false-negative, which are defined as:

$$\begin{aligned} TP_{i,soft} &= \sum_{k=1}^{p} \hat{y}_{ik} \cdot \mathbb{1}_{y_{ik}=1, \hat{y}_{ik} > th}(k) \\ FP_{i,soft} &= \sum_{k=1}^{p} r_{ik} \cdot \mathbb{1}_{y_{ik}=0, \hat{y}_{ik} > th}(k) \\ FN_{i,soft} &= \sum_{k=1}^{p} r_{ik} \cdot \mathbb{1}_{y_{ik}=1, \hat{y}_{ik} \leq th}(k), \end{aligned} \tag{9}$$

Note that in $FN_{i,soft}$ and $FP_{i,soft}$ we are adding up r_{ik} instead of $\hat{y}_{ik}$, because we measure how "false" those misclassified elements are.

To further exploit the information in misclassifications, following the same spirit of reformulation as in III-A3, the sum of residuals can be viewed as

$$\begin{aligned} \sum_{k=1}^{p} r_{ik} &= \sum_{k=1}^{p} \varphi(r_{ik}) \\ \varphi(x) &= x, \end{aligned} \tag{10}$$

which gives room for substituting the identical function $\varphi(x) = x$ with other more complicated ones. Considering the reformulation presented in III-A3, we can replace it with $\varphi(x) = -log(1-x)$ to get the final criterion function $F^{(i)}_{1,log}$, merged from cross-entropy loss and F_1 score, that is defined as:

$$\begin{aligned} F^{(i)}_{1,log} &= 2 \cdot \frac{precision_i \cdot recall_i}{precision_i + recall_i} \\ precision_i &= \frac{TP_{i,\varphi}}{TP_{i,\varphi} + FP_{i,\varphi}} \\ recall_i &= \frac{TP_{i,\varphi}}{TP_{i,\varphi} + FN_{i,\varphi}} \\ TP_{i,\varphi} &= \sum_{k=1}^{p} \hat{y}_{ik} \cdot \mathbb{1}_{y_i k=1, \hat{y}_{ik} > th}(k) \\ FP_{i,\varphi} &= \sum_{k=1}^{p} \varphi(r_{ik}) \cdot \mathbb{1}_{y_i k=0, \hat{y}_{ik} > th}(k) \\ FN_{i,\varphi} &= \sum_{k=1}^{p} \varphi(r_{ik}) \cdot \mathbb{1}_{y_i k=1, \hat{y}_{ik} \leq th}(k) \\ \varphi(x) &= -\log(1-x) \end{aligned} \tag{11}$$

We chose the log loss $\varphi(x) = -\log(1-x)$ used in cross-entropy over the mse loss ($\varphi(x) = x^2$) or ℓ_1 loss ($\varphi(x) = x$) because as the value of residual approaches 1, the log loss goes to infinity, which makes the function emphasize more on mistakenly classified elements and the noise packed in borderline misclassifications would be relatively suppressed.

IV. Experiments

To evaluate the effectiveness of the proposed reconstruction criterion, we combined $F^{(i)}_{1,log}$, $F^{(i)}_{1,elem}$ and $F^{(i)}_{1,soft}$ with both GRU-autoencoder and PCA to compare with traditional reconstruction criterion such as the cross-entropy loss. We also run experiments with classic anomaly detection methods,

TABLE II
AUC ROC scores for the GRU-autoencoder model combined with different criteria and trained with different subjects as normal data.

	1	2	3	4	5	6	7	8	9
cross entropy	0.975239	0.958159	0.974099	0.907872	0.764404	0.979918	0.985366	0.990756	**0.944715**
ℓ_2	0.964421	0.917244	0.972857	0.88324	0.697965	0.964851	0.97831	0.989469	0.925973
F_1	0.922404	0.886108	0.844787	0.766478	0.816244	0.925565	0.927039	0.979016	0.932907
$F_{1,soft}$	0.953818	0.928765	0.903571	0.854745	0.815791	0.968696	0.950684	0.993311	0.932291
$F_{1,log}$	**0.987227**	**0.985611**	**0.979339**	**0.954102**	**0.851323**	**0.999144**	**0.997623**	**0.996901**	0.944014
	10	11	12	13	14	15	16	17	Avg
cross entropy	0.981127	0.977195	0.951464	0.949133	0.941423	0.959979	0.89552	**0.937988**	0.94555
ℓ_2	0.979184	0.94489	0.896652	0.857217	0.881879	0.943517	0.861038	0.925543	0.916721
F_1	0.967334	0.977196	0.804858	0.815347	0.907183	0.945377	0.84815	0.746241	0.883073
$F_{1,soft}$	0.97372	0.987033	0.824479	0.841786	0.911925	0.954494	0.86734	0.773368	0.907989
$F_{1,log}$	**0.991274**	**0.993245**	**0.953315**	**0.969155**	**0.952176**	**0.981794**	**0.934214**	0.935161	**0.965036**

TABLE III
AUC ROC scores for the PCA model combined with different criteria and trained with different subjects as normal data.

	1	2	3	4	5	6	7	8	9
cross entropy	0.905623	0.932903	0.889032	0.874782	0.809761	0.917691	0.917097	0.923905	**0.953029**
ℓ_2	0.893629	0.869883	0.875814	0.815733	0.740041	0.888635	0.89731	0.908899	0.942671
F_1	0.902749	0.905896	0.88644	0.857147	0.861196	0.936294	0.976685	0.902188	0.921901
$F_{1,soft}$	0.904935	0.913003	0.889323	0.875949	0.871135	0.936922	**0.979903**	0.905859	0.924378
$F_{1,log}$	**0.921769**	**0.9533**	**0.903276**	**0.909348**	**0.876643**	**0.967007**	0.963049	**0.947192**	0.944199
	10	11	12	13	14	15	16	17	Avg
cross entropy	0.942259	0.94695	0.939047	0.907387	0.925588	0.874012	0.857456	0.872571	0.905241
ℓ_2	0.93422	0.932748	0.923125	0.865657	0.901959	0.843484	0.833475	0.837728	0.876765
F_1	0.943334	0.965608	0.916079	0.892141	0.931789	0.890752	0.839821	0.887454	0.90691
$F_{1,soft}$	0.943902	0.969193	0.920286	0.89813	0.935591	0.896031	0.844782	0.889675	0.911706
$F_{1,log}$	**0.962085**	**0.97575**	**0.955309**	**0.931534**	**0.952503**	**0.905517**	**0.884555**	**0.910486**	**0.933148**

TABLE IV
AUC ROC scores for different methods and criterion-model combination. Each column indicates that the corresponding subject is used as normal data. Here we denote GRU-autoencoder as GRU to save space in the table.

	1	2	3	4	5	6	7	8	9
LOF	0.652200	0.792872	0.877586	0.627958	0.905282	0.865176	0.669485	0.779208	0.711425
OCSVM	0.94865	0.870193	0.953574	0.79841	0.72567	0.945882	0.922988	0.983601	0.932072
ce (GRU)	0.975239	0.958159	0.974099	0.907872	0.764404	0.979918	0.985366	0.990756	0.944715
ce (PCA)	0.905623	0.932903	0.889032	0.874782	0.809761	0.917691	0.917097	0.923905	**0.953029**
$F_{1,log}$ (PCA)	0.921769	0.9533	0.903276	0.909348	**0.876643**	0.967007	0.963049	0.947192	0.944199
$F_{1,log}$ (GRU)	**0.987227**	**0.985611**	**0.979339**	**0.954102**	0.851323	**0.999144**	**0.997623**	**0.996901**	0.944014
	10	11	12	13	14	15	16	17	Avg
LOF	0.708235	0.646512	0.671088	0.646572	0.640881	0.692538	0.6220062	0.720277	0.719371
OCSVM	0.98864	0.913311	0.892281	0.882556	0.870536	0.847483	0.81662	0.929258	0.895396
ce (GRU)	0.981127	0.977195	0.951464	0.949133	0.941423	0.959979	0.89552	**0.937988**	0.94555
ce (PCA)	0.942259	0.94695	0.939047	0.907387	0.925588	0.874012	0.857456	0.872571	0.905241
$F_{1,log}$ (PCA)	0.962085	0.97575	0.955309	0.931534	0.952503	0.905517	0.884555	0.910486	0.933148
$F_{1,log}$ (GRU)	**0.991274**	**0.993245**	**0.953315**	**0.969155**	**0.952176**	**0.981794**	**0.934214**	0.935161	**0.965036**

including OC-SVM and Local outlier factor(LOF), as part of the baseline models.

A. Experimental Setup

Following the one-class classification setup, to test the performance of a specific algorithm, we train one model for each subject in the data set, that is, 17 models were trained for each algorithm. When training for subject A, for example, the data matrix of subject A was temporally divided into 3 consecutive parts, namely training, validating ,and testing. We would only use the training part of subject A to train the model, and use the test part of the data from subject A and other subjects combined as normal/anomalous test data respectively.

1) Models:

1) PCA: The crucial hyper-parameter in the PCA model is the number of features after projection. Here we use the explained variance ratio to determine the number of features to keep. As shown in figure 3, for most of the models, the explained variance can reach 90% with the first 60 dimensions. Thus we keep 60 dimensions (first 60 projected features).
2) GRU-autoencoder: Considering the similarity of PCA and GRU-autoencoder, i.e., both are reconstruction based methods, we borrow the information of PCA explained variance and chose the size of the hidden state in GRU network to be 60 as well. We chose cross-entropy as the training criterion. The model was trained

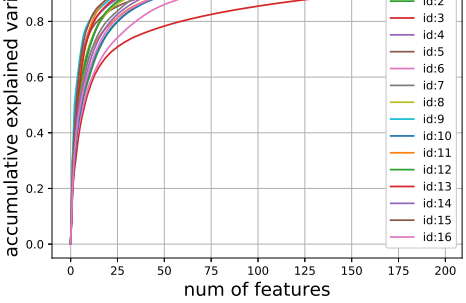


Fig. 3. Accumulative explained variance for PCA models trained with each subject as training data.

with Adam [5], in mini-batch style. The batch size is 128, and the learning rate is chosen to be 0.0015. Other hyper-parameters were set to follow the default set up in the Pytorch [8] package.

3) One-class SVM (OCSVM) [10] is a widely used one-class classification method, where a high dimensional sphere surrounding the training normal data points is found and used to predict whether a given test sample is anomalous or not. We used the Radial basis function (RBF) kernel as the non-linear kernel for the model.
4) Local outlier factor (LOF) [1] is a classic distance-based anomaly detection method, where a sample point is recognised as an anomaly when its local density is lower than its k nearest neighbours. In our experiment k is chosen to be 20.

2) Metrics: The traditional of comparing the performance of different classification methods would be demonstrating the ROC curve. Considering the fact that there are multiple models trained for each method, we mainly demonstrate the Area Under Curve (AUC) of ROC curves.

B. Results and Comparison

By examining table II and III, we can find that $F_{1,log}$ outperforms $F_{1,soft}$, and $F_{1,soft}$ in turn outperforms F_1. Thus proved the effectiveness of the proposed formulation based on F_1 score, as shown in equation 11. The formulation in the equation also gives possibilities of using other functions other than $\varphi(x) = -\log(1-x)$ or $\varphi(x) = x^2$ when encountering other sparse distributions.

Another observation is that the performance gain of using $F_{1,log}$ compared with cross-entropy loss is higher when combined with PCA than GRU-autoencoder. The information from table I, where we can find that the difference of ℓ_{ce}^{+} and ℓ_{ce}^{-} is more significant in the PCA model than in GRU-autoencoder, combined with table II and III also shows that the performance gain of our proposed criterion comes from better merging two parts of residual based loss.

We can also find that other than a few exceptions (2 out of 17 for both PCA and GRU-autoencoder), the $F_{1,log}$ score outperforms other criteria, which proved the effectiveness of using our proposed criterion in a sparse data set. As shown in table IV, the best performance can almost always be achieved when combining the proposed $F_{1,log}$ with GRU-autoencoder. In the few exceptions where the other methods do perform better, the margins are small (0.9379 vs 0.9351 and 0.9530 vs 0.9440). This gives a satisfying solution for user identity anomaly detection with behaviour information in the computer system.

V. Conclusion and Future Work

In this paper, we examined the class imbalance problem in reconstruction error based one-class classification problem, analysed the cause of it, and proposed a novel reconstruction criterion. The experiment shows that our criterion can effectively improve the performance of both PCA and GRU-autoencoder, and when combined with GRU-autoencoder, the model can serve as a practical usage behaviour based anomaly detection system. Regarding future work, more possible selections for the φ function can be further examined.

VI. Acknowledgment

This work was supported in part by NSF grant CNS-1814557 and MIT Lincoln Grant Air Force Contract No. A8721-05-C-0002 and FA8702-15-D-0001.

References

[1] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[2] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[3] Ville Hautamaki, Ismo Karkkainen, and Pasi Franti. Outlier detection using k-nearest neighbour graph. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 430–433. IEEE, 2004.

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

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

[6] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

[7] Cheng-Yuan Liou, Wei-Chen Cheng, Jiun-Wei Liou, and Daw-Ran Liou. Autoencoder for words. *Neurocomputing*, 139:84–96, 2014.

[8] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[9] Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, pages 4–11, 2014.

[10] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. Support vector method for novelty detection. In *Advances in neural information processing systems*, pages 582–588, 2000.

[11] David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54(1):45–66, 2004.

[12] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference*, pages 187–196, 2018.