Title

Online Dictionary Learning Based Fault and Cyber Attack Detection for Power Systems

Authors

Intriago, Gabriel
Zhang, Yu

Publication Date

2021-07-29

DOI

10.1109/pesgm46819.2021.9637891

Online Dictionary Learning Based Fault and Cyber Attack Detection for Power Systems

Gabriel Intriago and Yu Zhang
Department of Electrical and Computer Engineering
University of California, Santa Cruz
Emails: {gintriag, zhangy}@ucsc.edu

2021 IEEE Power & Energy Society General Meeting (PESGM) | 978-1-6654-0507-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/PESGM46819.2021.9637891

***Abstract*—The emerging wide area monitoring systems (WAMS) have brought significant improvements in electric grids' situational awareness. However, the newly introduced system can potentially increase the risk of cyber-attacks, which may be disguised as normal physical disturbances. This paper deals with the event and intrusion detection problem by leveraging a stream data mining classifier (Hoeffding adaptive tree) with semi-supervised learning techniques to distinguish cyber-attacks from regular system perturbations accurately. First, our proposed approach builds a dictionary by learning higher-level features from unlabeled data. Then, the labeled data are represented as sparse linear combinations of learned dictionary atoms. We capitalize on those sparse codes to train the online classifier along with efficient change detectors. We conduct numerical experiments with industrial control systems cyber-attack datasets. We consider five different scenarios: short-circuit faults, line maintenance, remote tripping command injection, relay setting change, as well as false data injection. The data are generated based on a modified IEEE 9-bus system. Simulation results show that our proposed approach outperforms the state-of-the-art method.**

I. Introduction

The ongoing improvements in wide-area monitoring systems have brought better visibility of the power system and have exposed the system to malicious cyber-attacks [1], [2]. In this context, event and intrusion detection systems (EIDS) are indispensable to classify the nature of a power system disturbance: is it a regular operation, fault condition, or a cyber-attack? The main challenge of this classification task is to extract relevant information from system measurements. Over the past decade, various data-driven techniques have been explored to tackle this problem.

A geometrical analysis of unsynchronized and synchronized attacks is introduced to detect the presence of attacks and identify compromised micro-PMUs [3]. Based on text-mining techniques, a data-driven approach was developed for false data attacks classification [4]. In recent years, classical machine learning algorithms, such as naive Bayes, support vector machines (SVM), and random forests (RF), have been applied to detect cyber-attacks and disturbances in power systems [5], [6], [7], [8]. It is also possible to build common paths of critical states by exploiting the relationships among voltage, current, and impedance to discover relevant patterns [9], [10]. Those classical methods, which often have difficulties dealing with large-scale and time-varying data, are unsuitable for real-time changing environments. Hence, stream data mining algorithms have recently drawn much attention. These include nonnested generalized exemplars [11], Hoeffding adaptive tree (HAT) [12], and HAT with change detectors [13]. Leveraging phasor measurement units (PMUs), those algorithms are proven to outperform classical methods in real systems. In [14], the authors proposed a transfer learning HAT model with one change detector, the adaptive sliding window (ADWIN). Their approach transferred knowledge from four datasets, where each dataset corresponds to a specific frequency oscillation.

It is challenging and costly to label a massive amount of PMU data on the fly in practice. Compared with data collection that depends only on data storage capacity, data labeling often requires rich domain knowledge of experts who can actively identify instances' labels. Therefore, we have abundant unlabeled data and scarce labeled data that share the same generative distribution. Due to this fact, semi-supervised learning (SSL) is an appropriate tool that combines a small amount of labeled data with a large amount of unlabeled data during training [15].

This paper proposes a novel approach for power system EIDS to improve the classification performance by transforming the data through higher-level representations extracted from an unlabeled dataset. In addition, we provide performance analysis for different sizes of the labeled dataset. To the best of our knowledge, this is the first effort to incorporate SSL with a stream data mining classifier for the EIDS. The rest of the paper is organized as follows. Section II presents the details of the proposed approach. Section III shows the simulation results. Finally, section IV gives the conclusion.

II. Semi-Supervised Hoeffding Adaptive Tree

We learn a dictionary by extracting higher-level features (such as oscillations, sudden changes, gradual changes, stable periods) from the unlabeled dataset to represent later the labeled data, which are then used to train a classifier incrementally.

This work was supported in part by a Seed Fund Award from CITRIS and the Banatao Institute at the University of California, and the Hellman Fellowship.

A. Online Dictionary Learning

Given a set of unlabeled instances $\mathcal{U} = \left\{\mathbf{x}_{\text{u}}^{(1)}, \ldots, \mathbf{x}_{\text{u}}^{(p)}\right\}$, where $\mathbf{x}_{\text{u}}^{(i)} \in \mathbb{R}^n$ is the i-th input feature vector, we formulate the following optimization problem to learn a new feature space representing these data points:

$$\underset{\mathbf{D}, \{\boldsymbol{\alpha}_{\text{u}}^{(i)}\}}{\text{minimize}} \quad \frac{1}{2}\sum_{i=1}^{p}\left\|\mathbf{x}_{\text{u}}^{(i)} - \mathbf{D}\boldsymbol{\alpha}_{\text{u}}^{(i)}\right\|_2^2 \tag{1a}$$

$$\text{subject to} \quad \left\|\boldsymbol{\alpha}_{\text{u}}^{(i)}\right\|_0 \leq k,\ i = 1, 2, \ldots, p, \tag{1b}$$

$$\|\boldsymbol{d}_j\|_2 \leq 1,\ j = 1, 2, \ldots, m \tag{1c}$$

The optimization variables are the dictionary $\mathbf{D}_t = \left[\mathbf{d}_1, \ldots, \mathbf{d}_m\right] \in \mathbb{R}^{n\times m}$ and the sparse codes $\boldsymbol{\alpha}_{\text{u}}^{(i)} \in \mathbb{R}^m, i = 1, 2, \ldots, p$. Typically, we have $m \gg n$ so that the dictionary is rich enough. Hence, by the least square objective, each input $\mathbf{x}_{\text{u}}^{(i)}$ is approximately represented as a linear combination of very few basis vectors in $\mathbf{D}$ with the corresponding coefficients given by $\boldsymbol{\alpha}_{\text{u}}^{(i)}$. The zero norm $\|\mathbf{a}\|_0$ denotes the number of non-zero coordinates of $\mathbf{a}$. Hence, the first constraint forces the vector $\boldsymbol{\alpha}_{\text{u}}^{(i)}$ to have at most k nonzero elements. The energy of each atom (basis) in the dictionary $\mathbf{D}$ is bounded by one, as given by the second constraint. This constraint prevents the entries of $\mathbf{D}$ from being arbitrarily large while the entries of $\boldsymbol{\alpha}_{\text{u}}^{(i)}$ being very small.

We leverage the alternating minimization method for the resulting nonconvex problem (1), i.e., minimizing one variable at each step while keeping all other variables fixed [16]. In the first step, we obtain the sparse codes $\boldsymbol{\alpha}_{\text{u}}^{(i)}, i = 1, 2 \ldots, p$. The second step updates the dictionary $\mathbf{D}$.

- *Sparse coding – optimization over* $\boldsymbol{\alpha}_{\text{u}}^{(i)}$: Start with a fixed random dictionary $\mathbf{D}$, and solve (1) with the orthogonal matching pursuit (OMP) algorithm to obtain the $\boldsymbol{\alpha}_{\text{u}}^{(i)}$ that corresponds to the unlabeled point $\mathbf{x}_{\text{u}}^{(i)}$ for $i = 1, 2, \ldots, p$.
- *Dictionary update – optimization over* $\mathbf{D}$: Keep $\{\boldsymbol{\alpha}_{\text{u}}^{(i)}\}_{i=1}^{p}$ fixed, find the dictionary $\mathbf{D}_t$ by sequentially updating each atom via the block-coordinate descent (BCD) algorithm:

$$\mathbf{u}_j = \mathbf{A}_{jj}^{-1}(\mathbf{b}_j - \mathbf{D}_{t-1}\mathbf{a}_j) + \mathbf{d}_j, \quad j = 1, 2, \ldots, m \tag{2}$$

$$\mathbf{d}_j = \frac{\mathbf{u}_j}{\max(\|\mathbf{u}_j\|_2, 1)}, \quad j = 1, 2, \ldots, m, \tag{3}$$

where $\mathbf{D}_{t-1}$ is the dictionary at the previous iteration. The matrices $\mathbf{A} = \left[\mathbf{a}_1, \ldots, \mathbf{a}_m\right] = \boldsymbol{\alpha}_{\text{u}}^{(i)}\boldsymbol{\alpha}_{\text{u}}^{(i)^\top} \in \mathbb{R}^{m\times m}$ and $\mathbf{B} = \left[\mathbf{b}_1, \ldots, \mathbf{b}_m\right] = \mathbf{x}_{\text{u}}^{(i)}\boldsymbol{\alpha}_{\text{u}}^{(i)^\top} \in \mathbb{R}^{n\times m}$ carry the information of the updated $\boldsymbol{\alpha}_{\text{u}}^{(i)}$'s. The update repeats until $\mathbf{D}_t$ converges.

B. New Feature Representation

Consider a set of labeled instances $\mathcal{L} = \left\{(\mathbf{x}_\ell^{(1)}, y^{(1)}), \ldots, (\mathbf{x}_\ell^{(q)}, y^{(q)})\right\}$, where $\mathbf{x}_\ell^{(i)} \in \mathbb{R}^n$ is the ℓ-th input feature vector with label $y^{(i)} \in \{1, \ldots, C\}$. Upon learning the dictionary $\mathbf{D}^*$ as elaborated above, the labeled data can be represented by using the basis vectors of $\mathbf{D}$. This is carried out by solving the following problem via the OMP for each labeled data point:

$$\underset{\boldsymbol{\alpha}_\ell^{(i)}}{\text{minimize}} \quad \frac{1}{2}\left\|\mathbf{x}_\ell^{(i)} - \mathbf{D}^*\boldsymbol{\alpha}_\ell^{(i)}\right\|_2^2 \tag{4a}$$

$$\text{subject to} \quad \left\|\boldsymbol{\alpha}_\ell^{(i)}\right\|_0 \leq k. \tag{4b}$$

In other words, a labeled data point is now approximately represented as a linear combination of the learned atoms as:

$$\mathbf{x}_\ell^{(i)} = \mathbf{D}\boldsymbol{\alpha}_\ell^{(i)} + \boldsymbol{\eta}, \tag{5}$$

where $\boldsymbol{\eta}$ is the reconstruction error. We preserve each original label of $y^{(i)}$ by attaching it to the new representation; i.e., the k-sparse code $\boldsymbol{\alpha}_\ell^{(i)}$ in a higher dimensional space. Finally, we train the HAD classifier with these new representations by using the software package MOA [17].

Algorithm 1 Semi-supervised HAD (SSHAD)

Require:
1) Unlabeled data $\mathcal{U} = \{\mathbf{x}_{\text{u}}^{(1)}, \ldots, \mathbf{x}_{\text{u}}^{(p)}\}$
2) Labeled data $\mathcal{L} = \left\{(\mathbf{x}_\ell^{(1)}, y^{(1)}), \ldots (\mathbf{x}_\ell^{(q)}, y^{(q)})\right\}$.
3) Randomly initialize $\mathbf{D}$ from unlabeled data vectors.
4) Maximum iteration: max_iter $= 200$.

1: Normalize the labeled and unlabeled data.
2: **for** $t = 1$ to max_iter **do**
3: Compute the sparse code $\boldsymbol{\alpha}_{\text{u}}$ with $\mathbf{D}_{t-1}$ by solving (1).
4: Update $\mathbf{D}_t$ keeping the matrix $\boldsymbol{\alpha}_u$ fixed.
5: **end for**
6: Solve (4b) to obtain the matrix $\boldsymbol{\alpha}_\ell$.
7: Attach to $\boldsymbol{\alpha}_\ell$ the labels from $\mathbf{x}_\ell$.
8: Train HAD with the new labeled dataset $\hat{\mathcal{L}} = \left\{(\boldsymbol{\alpha}_\ell^{(1)}, y^{(1)}), \ldots, (\boldsymbol{\alpha}_\ell^{(q)}, y^{(q)})\right\}$ using MOA.
9: **return** The trained HAD classifier.

Remark (Matching pursuit vis-a-vis LASSO)**.** *The sparse dictionary learning problem generally has two different formulations: matching pursuit and LASSO. The former is shown by problem (1) while the latter is relaxing ℓ_0 norm to ℓ_1 norm and being lifting to the objective as a soft constraint. The matching pursuit formulation explicitly guarantees k-sparsity, which is more user-friendly to find the "best" value of k by trial-and-error simulations. According to our numerical experiments that will be discussed in the next section, we find that the solution to the matching pursuit is more stable numerically.*

Algorithm 1 features two essential differences from the algorithm in [18]. In [18], the authors build the dictionary using self-taught learning (unlabeled and labeled datasets have different generative distributions [19]) to later train and test an SVM classifier with the new representation of the labeled dataset. In contrast, our model builds the dictionary using SSL and next incrementally trains a HAD classifier with all the transformed labeled dataset instances. In a nutshell,

our algorithm capitalizes on semi-supervised knowledge to enhance the HAD classifier's overall performance. We name the proposed algorithm as SSHAD, where "SS" stands for semi-supervised, to differentiate it from the original version of HAD presented in [13].

C. HAD Classifier

HAD is composed of three main ingredients: a window to remember recent examples, a distribution-change detector, and an estimator for some statistics of the input data. Once a change is detected, an alternate tree will be created and grow with the instances appearing right after the change. The existing alternate tree will replace the current tree if it is more accurate. The HAT [20] is the parent tree of HAD, where the former has only one change detector, ADWIN, whereas HAD has two change detectors ADWIN and DDM.

ADWIN serves as an estimator and change detector that keeps a variable-length window $\mathcal{W}$ of recent data such that the window has the maximal length statistically consistent with the null hypothesis of the average value inside the window has not changed. When two "big enough" sub-windows of $\mathcal{W}$ have "distinct enough" averages, it can be said with high probability that a change in the data distribution has occurred and the older items in $\mathcal{W}$ should be dropped. The "big and distinct enough" can be quantitatively defined by the Hoeffding bound [21].

DDM is a change detector that relies on the concept of 'context' defined as a set of contiguous examples whose data distribution is stationary. DDM incrementally controls the error rate of the model. Statistical theory guarantees that the error decreases if the data distribution remains stationary, and error increases when the distribution changes. A new context is declared if the error reaches a warning level at instance k_w and a drift level at instance k_d. Given that, this indicates a distribution change, and a new model is learned by using the examples between k_w and k_d. A detailed explanation of DDM can be found in [22].

III. Experiments and Results

A. Datasets

Power system attack datasets [23] are used to test the performance of our proposed approach. There are three datasets: 2-class, 3-class, and 37-class datasets, where each of them includes 128 features split into two categories: physical (voltages, currents, and impedances) and cyber-physical (control logs, network alerts, and relay logs) features. Five scenarios are considered: short-circuit faults, line maintenance, remote tripping command injection (attack), relay setting change (attack), as well as data injection (attack). Fig. 5 shows the testbed architecture used in generating the datasets.

B. Implementation and Parameters

We run all the experiments using MATLAB, WEKA, and the massive online analysis (MOA) software [24]. The relevant parameters were obtained by using cross-validation. The value of max_iter $= 200$ yielded best results. The parameter k was set to 10 for both OMP procedures, i.e, each of $\boldsymbol{\alpha}_u^{(i)}$'s and

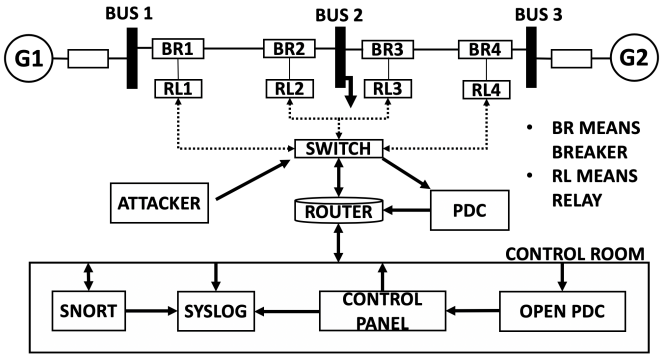

Fig. 1. Test-bed architecture for generating the datasets

$\boldsymbol{\alpha}_\ell^{(i)}$'s has at most ten nonzero values. We tested different sizes for the dictionary and found that 130 atoms performed the best. For both OMP optimization problems, the tolerance of the squared ℓ_2-norm residual was set to 0.01. Finally, the parameters for the HAD were set to the default values given by MOA.

C. Performance Metrics

In this work, we used the prequential evaluation technique, where each instance is used to test and then train the model. Because of this online setup, the accuracy is incrementally updated. We chose the classification accuracy, the Kappa statistic, evaluation time, and model cost to evaluate our approach's performance; see also [13]. The Kappa statistic is a measure for rating classification accuracy for imbalance scenarios in offline and online classification. The Kappa statistic is defined as:

$$\kappa = \frac{\rho_o - \rho_e}{1 - \rho_e}, \tag{6}$$

where ρ_o is the accuracy of the classifier under analysis, and ρ_e is the accuracy of a random classifier. If the classifier predicts all the time correctly, $\kappa = 1$. If the classifier performs like a random classifier, $\kappa = 0$. The evaluation time consists of both training and testing time because there is no clear separation between them [13]. The model cost is measured in RAM per hour (hereafter referred to as Ram-Hours) [12].

D. Simulation Results

We conduct classification experiments using the 2-class, 3-class, and 37-class datasets. The performance results were obtained with five different sizes, determined by the labeled dataset's sampling ratio. All values given in figures and tables are 10-fold average. The performance of our model improves with the increased size of the unlabeled dataset. It can be seen that the performance gets saturated with 50,000 unlabeled data points.

Fig. 2, 3 and Tab. I show the classification results for the 2-class and 3-class datasets. It can be seen that the performances of SSHAD and HAD are similar. However, when it comes to the 37-class dataset, our model clearly outperforms HAD as shown in Fig. 6, 4 and Tab. II. These results corroborate

the merits of our proposed approach, representing the data by higher-level features yields more accurate identification of events in power systems. Moreover, as shown in Fig. 5, SSHAD is robust to the presence of bad data.

TABLE I
THE 3-CLASS DATASET: 10-FOLD AVERAGE KAPPA ($\overline{\kappa}$) AND COST ($\overline{cost}$) COMPARISONS BETWEEN SSHAD (k = 10) AND HAD.

Sampling Ratio	$\overline{\kappa}(\%)$		$\overline{cost}$ (Ram-Hour)	
	SSHAD	HAD	SSHAD	HAD
10%	82.25	82.29	1.43×10^{-8}	1.37×10^{-8}
30%	88.36	88.44	2.18×10^{-8}	2.07×10^{-8}
50%	**69.87**	69.57	$\mathbf{2.87 \times 10^{-8}}$	2.89×10^{-8}
70%	**59.56**	59.28	$\mathbf{3.67 \times 10^{-8}}$	3.83×10^{-8}
90%	**51.91**	51.70	$\mathbf{4.63 \times 10^{-8}}$	5.27×10^{-8}

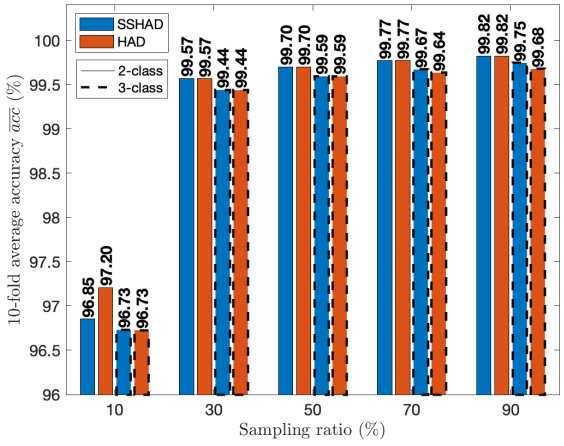

Fig. 2. 10-fold average accuracy ($\overline{acc}$) comparison between SSHAD with parameter k = 10 and HAD using the 2-class and 3-class datasets.

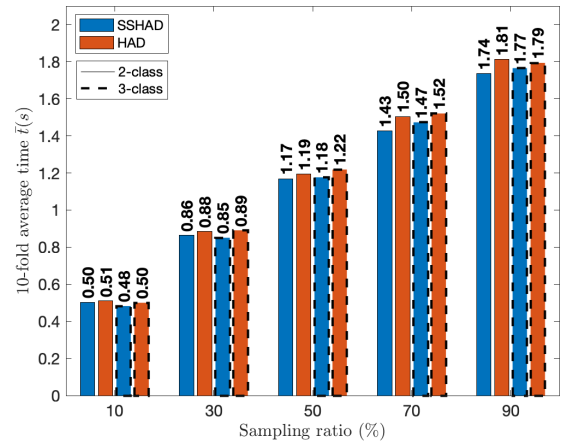

Fig. 3. 10-fold average accuracy ($\overline{acc}$) comparison between SSHAD with parameter k = 10 and HAD using the 2-class and 3-class datasets.

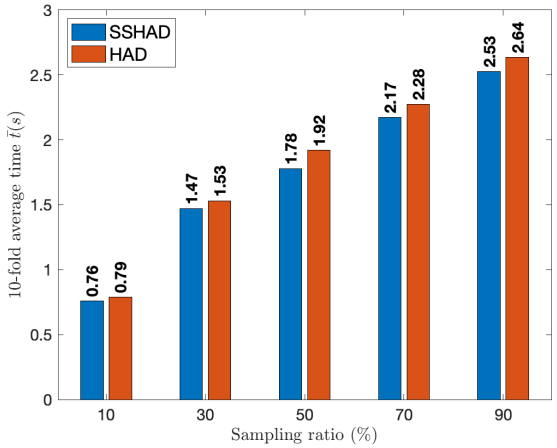

Fig. 4. 10-fold average time ($\overline{t}$) comparison between SSHAD with parameter k = 10 and HAD using the 37-class dataset.

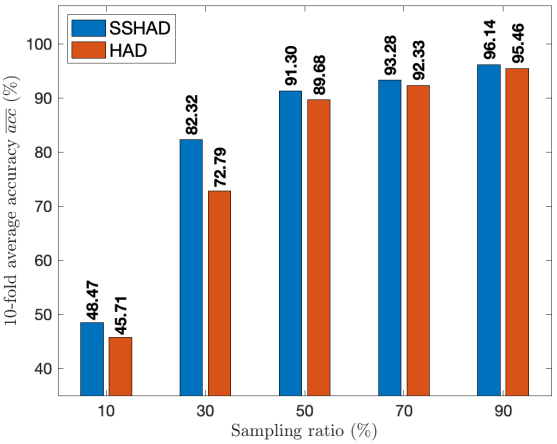

Fig. 5. 10-fold average time ($\overline{t}$) comparison between SSHAD with parameter k = 10 and HAD using the 37-class dataset in the presence of 10% of bad data.

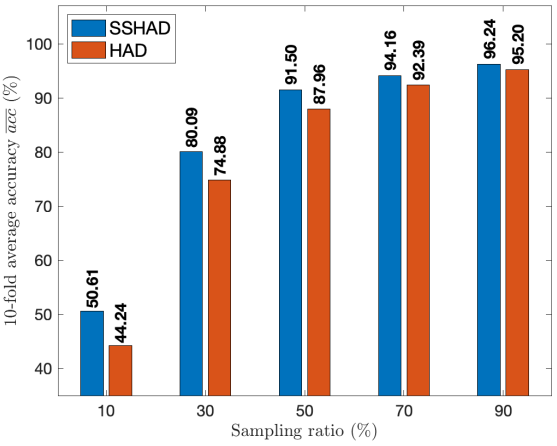

Fig. 6. 10-fold average accuracy ($\overline{acc}$) comparison between SSHAD with parameter k = 10 and HAD using the 37-class dataset.

TABLE II
THE 37-CLASS DATASET: 10-FOLD AVERAGE KAPPA ($\overline{\kappa}$) AND COST ($\overline{cost}$) COMPARISONS BETWEEN SSHAD (k = 10) AND HAD.

Sampling Ratio	$\overline{\kappa}(\%)$		$\overline{cost}$ (Ram-Hour)	
	SSHAD	HAD	SSHAD	HAD
10%	**29.80**	28.39	$\mathbf{5.80 \times 10^{-8}}$	5.88×10^{-8}
30%	**69.62**	62.32	$\mathbf{1.20 \times 10^{-7}}$	1.64×10^{-7}
50%	**79.48**	77.41	$\mathbf{9.24 \times 10^{-8}}$	1.09×10^{-7}
70%	**82.33**	80.30	$\mathbf{1.09 \times 10^{-7}}$	1.13×10^{-7}
90%	**85.64**	84.38	$\mathbf{1.24 \times 10^{-7}}$	1.28×10^{-7}

IV. CONCLUSION

We develop a semi-supervised online approach (SSHAD) for the power system event detection in this paper. The labeling process for a large amount of unlabeled data is often very time-consuming and costly, requiring specific domain knowledge of many experts. Considering this fact, we leverage online dictionary learning techniques to automatically build a new feature space for the labeled data examples by extracting valuable information from the unlabeled dataset. The learned sparse codes of the labeled instances become the new feature representations, based on which we train the HAD classifier.

Extensive numerical results corroborate our proposed approach's effectiveness that yields a better classification performance and compensates for the additional computational burden of learning the higher dimensional representations. Despite these results, we acknowledge that future work is needed to make our approach more robust. For instance, this work can be extended by studying how a malicious adversary can modify the data and determining the depth of its attack from the game theory perspective. Finally, a more detailed analysis of the temporal dependence of the data should be considered.

REFERENCES

[1] S. Ntalampiras, "Detection of integrity attacks in cyber-physical critical infrastructures using ensemble modeling," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 1, pp. 104–111, Feb. 2015. [Online]. Available: https://doi.org/10.1109/tii.2014.2367322

[2] H. Lin, Y. Deng, S. Shukla, J. Thorp, and L. Mili, "Cyber security impacts on all-PMU state estimator - a case study on co-simulation platform GECO," in *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, Nov. 2012. [Online]. Available: https://doi.org/10.1109/smartgridcomm.2012.6486049

[3] M. Kamal, M. Farajollahi, H. Nazaripouya, and H. Mohsenian-Rad, "Cyberattacks against event-based analysis in micro-PMUs: Attack models and counter measures," *IEEE Transactions on Smart Grid*, pp. 1–1, 2020. [Online]. Available: https://doi.org/10.1109/tsg.2020.3029937

[4] R. Ma, S. Basumallik, and S. Eftekharnejad, "A PMU-based data-driven approach for classifying power system events considering cyberattacks," *IEEE Systems Journal*, vol. 14, no. 3, pp. 3558–3569, Sep. 2020. [Online]. Available: https://doi.org/10.1109/jsyst.2019.2963546

[5] R. C. B. Hink, J. M. Beaver, M. A. Buckner, T. Morris, U. Adhikari, and S. Pan, "Machine learning for power system disturbance and cyber-attack discrimination," in *2014 7th International Symposium on Resilient Control Systems (ISRCS)*. IEEE, Aug. 2014. [Online]. Available: https://doi.org/10.1109/isrcs.2014.6900095

[6] M. A. Karim, M. Chenine, K. Zhu, L. Nordstrom, and L. Nordström, "Synchrophasor-based data mining for power system fault analysis," in *2012 3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*. IEEE, Oct. 2012. [Online]. Available: https://doi.org/10.1109/isgteurope.2012.6465843

[7] K. Demertzis and L. Iliadis, "A computational intelligence system identifying cyber-attacks on smart energy grids," in *Springer Optimization and Its Applications*. Springer International Publishing, 2018, pp. 97–116. [Online]. Available: https://doi.org/10.1007/978-3-319-74325-7_5

[8] D. Wang, X. Wang, Y. Zhang, and L. Jin, "Detection of power grid disturbances and cyber-attacks based on machine learning," *Journal of Information Security and Applications*, vol. 46, pp. 42–52, Jun. 2019. [Online]. Available: https://doi.org/10.1016/j.jisa.2019.02.008

[9] S. Pan, T. Morris, and U. Adhikari, "A specification-based intrusion detection framework for cyber-physical environment in electric power system," *International Journal of Network Security*, vol. 17, pp. 174–188, 01 2015.

[10] ——, "Classification of disturbances and cyber-attacks in power systems using heterogeneous time-synchronized data," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 650–662, Jun. 2015. [Online]. Available: https://doi.org/10.1109/tii.2015.2420951

[11] U. Adhikari, T. H. Morris, and S. Pan, "Applying non-nested generalized exemplars classification for cyber-power event and intrusion detection," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 3928–3941, Sep. 2018. [Online]. Available: https://doi.org/10.1109/tsg.2016.2642787

[12] N. Dahal, O. Abuomar, R. King, and V. Madani, "Event stream processing for improved situational awareness in the smart grid," *Expert Systems with Applications*, vol. 42, no. 20, pp. 6853–6863, Nov. 2015. [Online]. Available: https://doi.org/10.1016/j.eswa.2015.05.003

[13] U. Adhikari, T. H. Morris, and S. Pan, "Applying hoeffding adaptive trees for real-time cyber-power event and intrusion classification," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 4049–4060, Sep. 2018. [Online]. Available: https://doi.org/10.1109/tsg.2017.2647778

[14] Z. E. Mrabet, D. F. Selvaraj, and P. Ranganathan, "Adaptive hoeffding tree with transfer learning for streaming synchrophasor data sets," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec. 2019. [Online]. Available: https://doi.org/10.1109/bigdata47090.2019.9005720

[15] K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell, *Machine Learning*, vol. 39, no. 2/3, pp. 103–134, 2000. [Online]. Available: https://doi.org/10.1023/a:1007692713085

[16] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*. ACM Press, 2009. [Online]. Available: https://doi.org/10.1145/1553374.1553463

[17] A. Bifet, R. Gavaldà, G. Holmes, and B. Pfahringer, *Machine Learning for Data Streams*. The MIT Press, 2018. [Online]. Available: https://doi.org/10.7551/mitpress/10654.001.0001

[18] F. Liu, J. Ma, R. Zhao, and Q. Wang, "Online dictionary self-taught learning for hyperspectral image classification," in *2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*. IEEE, May 2018. [Online]. Available: https://doi.org/10.1109/i2mtc.2018.8409676

[19] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning," in *Proceedings of the 24th international conference on Machine learning - ICML '07*. ACM Press, 2007. [Online]. Available: https://doi.org/10.1145/1273496.1273592

[20] A. Bifet and R. Gavaldà, "Adaptive learning from evolving data streams," in *Advances in Intelligent Data Analysis VIII*. Springer Berlin Heidelberg, 2009, pp. 249–260. [Online]. Available: https://doi.org/10.1007/978-3-642-03915-7_22

[21] ——, "Learning from time-changing data with adaptive windowing," in *Proceedings of the 2007 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, Apr. 2007. [Online]. Available: https://doi.org/10.1137/1.9781611972771.42

[22] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Advances in Artificial Intelligence – SBIA 2004*. Springer Berlin Heidelberg, 2004, pp. 286–295. [Online]. Available: https://doi.org/10.1007/978-3-540-28645-5_29

[23] U. Adhikari, S. Pan, T. H. Morris, and J. Beave, "Industrial control system (ics) cyber attack datasets, dataset 1: Power system datasets," https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets, accessed: 2020-08-11.

[24] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "Moa: Massive online analysis," *J. Mach. Learn. Res.*, vol. 11, p. 1601–1604, Aug. 2010.