Hierarchical Convolutional Neural Networks for Event Classification on PMU Measurements

Martin Pavlovski, *Student Member, IEEE*, Mohammad Alqudah, *Student Member, IEEE*, Tatjana Dokic, *Member, IEEE*, Ameen Abdel Hai, *Student Member, IEEE*, Mladen Kezunovic, *Life Fellow, IEEE*, Zoran Obradovic, *Senior Member, IEEE*

***Abstract*—Event classification is one of the central components of automated disturbance analysis based on PMU measurements. Obtaining high-quality *event labels* remains a challenge for supervised learning-based classification of local and system-wide events in power grids due to its labor-intensive requirement. We present a sensitivity study considering rapidly refined, partially and fully inspected event labels that leads to evidence that hierarchical convolutional neural networks (HCNNs) outperform traditional classification models regardless of the quality of the available event labels. It is demonstrated that performance similar to the one obtained using entirely domain-driven labeling can be achieved as long as the involved expert does not mislabel more than ~5% of the event data captured by PMU measurements.**

***Index Terms*—Power system events, Situational awareness, Machine learning, Convolutional neural networks.**

I. Introduction

A VARIETY of events are continually impacting the reliability of electric power systems. These events may be of different nature and type, caused by various factors, and typically occur at irregular time intervals. For instance, system-wide fundamental frequency events are quite different in nature as compared to localized transmission line faults. Measurements of electrical quantities characteristic to the behavior of a power system can be leveraged to analyze such events and potentially mitigate their impact on the system and its customers. In particular, patterns in measurements taken by sparsely located phasor measurement units (PMUs) can be analyzed to help understand and characterize local and system-wide events occurring across the system. With the recent deployment of a large number of PMUs across the U.S. power grid, and considering that PMU measurements are taken continuously in real time, an immense amount of PMU streaming data can be collected. This makes manual event analysis from PMU data practically infeasible.

This paper focuses on a *data-driven* approach to automated detection and further characterization of *local* and *system-wide* events from multiple, sparsely located *PMU measurements*. Such an approach can serve in performing data-driven disturbance monitoring without requiring details of the underlying physical model of the power system. While having well-defined *event labels* (event type indicators) describing PMU measurements helps in detecting and analyzing events, inspecting the measurements for potential event occurrences is labor-intensive and hence too costly. An additional challenge is posed when the physical model of the power system is unavailable, meaning that any event analysis would need to be carried out solely based on sparsely located PMU measurements. Thus, automated *data-driven* solutions to storing, processing and analyzing PMU measurements that do not require extensive labeling, as proposed in our study, are essential to effective event detection and classification.

To leverage the value of PMU measurements, many well-known predictive techniques have been utilized for automated event detection and fault analysis [1]. One line of research focuses on methods for extracting features from raw PMU measurements, which can be categorized in three main groups: 1) signal transform-based methods, such as Discrete Fourier Transform [2]-[3], Wavelet Transforms [2]-[4], and Fast Discrete S-Transform [2], [3], [5]; 2) linear projection methods, such as Principal Component Analysis [6] and Minimum Volume Enclosing Ellipsoid [7]; and 3) time series methods, such as domain-specific Shapelets [2]-[3], statistical feature extraction [8], and Detrended Fluctuation Analysis [9].

Having features extracted from PMU measurements, the subsequent event classification can be formulated as a classical machine learning problem and has been approached by a number of methods including Agglomerative Hierarchical Clustering [7], Extreme Learning Machines [5], K-Nearest Neighbor [2], [3], [10], Support Vector Machines [2], [3], [10], Decision Trees [10] and, more recently, Convolutional Neural Networks (CNNs) [4], [11]-[13]. CNNs have been successfully applied for detecting and classifying events in other industrial applications such as online motor bearing fault detection and diagnosis [14], [15], which has shown to be beneficial to equipment condition maintenance. Hierarchical CNNs were also used for detecting multiple fault conditions and estimating fault severities at the same time [16]. In the realm of power systems, a CNN variant was designed for event detection in nonintrusive load monitoring (NILM) applications [17] and for model identification, event detection, and topology change location based on distribution-level PMU measurements [18]. Although a very recent study [19] has focused on PMU-based detection of system-wide frequency events alone, leveraging

This material is based upon work supported by the Department of Energy under Award Number DE-OE0000913.

M. Pavlovski, M. Alqudah, A. A. Hai and Z. Obradovic are with Temple University, Philadelphia, PA 19122 USA (e-mails: martin.pavlovski@temple.edu; mohammad.alqudah@temple.edu; aabdelhai@temple.edu; zoran.obradovic@temple.edu). T. Dokic and M. Kezunovic are with Texas A&M University, College Station, TX 77843-3128 USA (e-mails: tatjana.djokic@tamu.edu; kezunov@ece.tamu.edu).

CNN variants to detect and further characterize local and system-wide events from measurements captured by sparsely located PMUs has not been addressed yet in the existing literature. CNNs applied to other event classification tasks typically approach the multiclass classification problem directly instead of breaking it down into multiple, less challenging, problems. In most cases, the available PMU-measured quantities are typically underutilized. Although several studies [20, 21, 22] discuss the utilization of different PMU quantities and the correlations among them, CNN models are capable of capturing such correlations by design.

Another line of research focuses on event labeling as a pivotal prerequisite for event classification. Multiple event classes including generator loss, line fault, load switching, and series capacitor switching were analyzed in [7]. Reactive power switching and synchronous motor switching in addition to the other three classes (fault, generation loss, load switching) were considered in [2]. Normal operation, malfunctioned capacitor bank switching and regulator on-load tap changer switching were reported in [23]. Normal operation and events representing voltage disturbance, voltage sag, motor start, and high impedance fault were studied in [24]. Up to 13 event classes were distinguished by utilizing unsupervised clustering algorithms, and then characterized by performing supervised classification based on the identified categories in [25].

In this study, the event classification problem is formulated from a measurement analysis perspective. In that regard, we take a data-driven approach to automate the extraction and analysis of information from reported PMU measurements with the objective of detecting and further classifying power system events. Our contribution is the utilization of end-to-end deep learning based convolutional neural network (CNN) variants for classification of local and system-wide events. Using the model variants we demonstrate the following benefits: 1) automatic feature learning without the need of any feature engineering interventions, 2) ability to utilize all measured PMU quantities, as well as each quantity separately, and 3) superior performance over traditional models when only a small fraction of labeled PMU measurements is utilized.

The paper first gives the background of the measurement problem we are addressing. The data and event classification models using the variants of CNN are discussed next. Then, the experimental study results are elaborated leading to the conclusions about the benefits of the leveraged model variants. The list of references is given at the end.

II. Background

A. Event Classification

In our case, events are instances of power system disturbances that are captured by PMU measurements. The objective of event classification is to identify the type (*class*) of an event once it appears in a measurement time interval. Event classification models are generally based either on *simulated data from physical models* of power grids or on *measurements* (*data-driven models*) collected from the grid [3]. In our study, neither the grid topology nor the geographical locations of the PMUs were made available by the data provider due to security and confidentiality constraints. This makes the simulation of a power system behavior involving different types of events infeasible. Hence, a data-driven approach is taken to indirectly infer the behavior of the physical model in situations of normal operation, and local and system-wide event occurrences by automatically processing PMU measurements. Unlike a physical model, a data-driven model learns from 'examples' of event occurrences. The time intervals and descriptions of event occurrences are recorded as an *event log*. As discussed in Section III-C, each record in an event log contains an *event label* that represents the type of the recorded event. The objective of a data-driven event classification model is to automatically learn the relationship between a collection of PMU measurements and their event labels recorded in a corresponding event log. For this purpose, any generic machine learning classifier can be incorporated in the event classification pipeline, illustrated in Fig. 1.

B. Challenges of Data-Driven Event Classification

Measurement-related challenges. There are *two* major challenges concerning the PMU measurements in this study.

1) Measurement quality. To inspect the quality of the measurements, we evaluated the status bits for the PMUs which were made available by the data provider. Status bits are 16-bit fields reserved for various flags that report the status of a PMU as device and communication network problems that may have resulted in corrupted or substandard measurements. However, given the lack of some critical information, such as the user settings for the PMUs, and compliance of the PMUs to a given version of the PMU standard, the information contained in the status bits was confirmed to be unusable. As a result, the measurement quality was assessed in the context of the analysis feasible for the development of a data-driven model. This process included filtering missing, duplicated, noisy and measurements with unreasonable scales, directly assessed from the reported measurements in the provided dataset. Specifically, it was observed that the fractions of missing values in voltage/current magnitude and frequency measurements of a single PMU may range from 0.69% to 30.01%. Across all PMUs, the fraction of missing values is 3.03%. The duplicate values found in the measurements were reported by a third of the PMUs (14 out of 43). The number of duplicates varies across different PMUs and time periods, but may reach up to ~4.63 hours of duplicate measurements on certain days. Other inconsistencies were also observed in ~12% of the PMUs (5 out of 43), including unreasonable voltage levels (concluded by analyzing the scale of the measurements), constant frequency measurements of 60 Hz over time, and inaccurate timestamps. Among those five PMUs, two generated measurements containing 2-4% noisy samples, while the other three had a

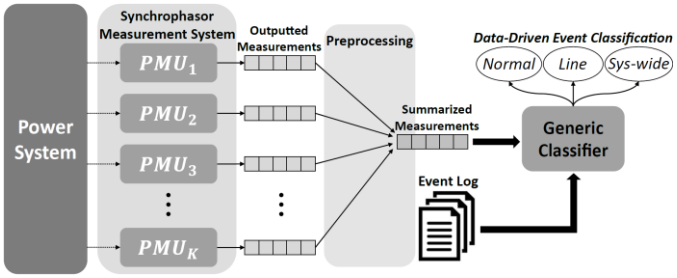

Fig. 1. Data-driven event classification pipeline.

significantly lower fraction of noisy values. Through visual inspection, a domain expert observed that some measurement quality was improved over the years. The accuracy of the measurements that were not observed to be unreasonable or noisy cannot be verified as all PMU measurements were received from a third-party data provider, and it is assumed that the measurement accuracies are compliant with the PMU standard [26]; since the goal of this study is not to characterize the measurements or their accuracy, but to automatically recognize (detect and classify) events using measurements taken with an assumed (standard-compliant) accuracy.

2) Scarce, imprecise, or unreliable event labels. Even with a set of high-precision and low-noise PMU measurements, building a data-driven event classification model requires an *event log* containing *event labels* associated with the measurements. Typically, PMU measurements are *scarcely* labeled as a large fraction of the events might not have been assigned event labels. In such cases, one solution is to label events by utilizing clustering techniques [7], [25] to automatically categorize the PMU measurements into a certain number of groups, each group supposedly representing a separate event type. The issue with this approach is the lack of domain knowledge creating the difficulty to associate the detected labels with certain types of well-known disruptive events. In addition, even when some event labels are provided, they are often *imprecise* or *unreliable*. For instance, the event log used in this study was provided by multiple data contributors and created with a combination of automatic event labeling and labeling based on subsequent investigation. The events were triggered by conditions including high/low voltage, high zero-phase current, generator breakers opening/closing, etc. The triggered events varied greatly among the data contributors and were recorded by instruments such as digital fault recorders (DFRs), digital relays, and PMUs. The events were initially labeled by operators that used the time reference from supervisory control and data acquisition (SCADA) systems and associated software tools; which is considerably less accurate that the time-tag information from the PMUs and their embedded GPS receivers, resulting in *imprecise* event labels. In some cases, the timestamps had to be entered manually into the event log, thus there is a considerable likelihood of human error, making some recorded event labels *unreliable*. As a solution, an additional domain expert is needed to help assign more precise labels by defining hard-coded rules [28]-[29] or visually inspecting a portion of the events that can be used to build a data-driven event classification model. While this approach incurs additional cost of a domain expert, it is expected to yield more reliable event labels.

Data-model related challenges. Despite the modeling advances in event classification from PMU measurements, existing studies seem to suffer from the following limitations:

Model capacity. As discussed in Section II-A, a generic data-driven classifier needs to be built solely based on the PMU measurements. The challenge of classifying events in such a scenario is to infer event patters from the PMU measurements while having a notion of neither the physical model of the underlying power system nor the specifics of the measurement system. The majority of previous studies on data-driven event classification are based on traditional machine learning models that face challenges related to high dimensionality, autocorrelation, and above all require features to be handcrafted or extracted from PMU measurements in advance. In either case, the classification performance depends on the method (e.g., a specific transform) used to engineer the features derived from the PMU measurements. As a result, model design based on automatic feature learning is essential towards attaining a more versatile and accurate event classification [4].

Local and system-wide event characterization. A plethora of studies [6], [8], [9], [19], [30] utilize models limited to conventional event detection from PMU measurements and are not capable of further classifying events into different categories once detected. The studies that extend this problem to event classification seem to either consider a more specific categorization of events [2], [7] or focus on subtypes of a major event type [12], [25], [29]. A more general categorization suggests that events may be localized once they occur (e.g., line faults), or have a system-wide manifestation (e.g., fundamental frequency events). Considering the geographical sparsity of PMU devices across the U.S. power system, covering less than 5% of the system's electrical buses, the odds of a local event occurring in the close proximity of a PMU are rather small. This poses an additional challenge of characterizing local events based on sparse measurements from PMUs that might be located away from the events' occurrence points.

III. Data

A. PMU Measurements

The measurements used in this study were reported by PMUs placed sparsely at unknown locations across the Western Interconnection of the U.S. power grid with over 20,000 buses. The measurements were collected over a two-year period (covering 2016 and 2017) in a dataset that amounts to the size of 7 TB. As mentioned earlier, neither the geographical locations of the PMUs nor the grid topology was made available by the data provider. For each PMU, multiple quantities were recorded including positive sequence (PS) voltage magnitude, PS current magnitude and system frequency, among others. The first two will be referred to as 'voltage' and 'current' for the sake of brevity. The measurements were collected over the two-year period (covering 2016 and 2017) at reporting rates of 30 and 60 frames per second (fps). The calculated PMU-measured quantities were assigned timestamps in UTC.

B. Measurement Preprocessing

The provided PMU dataset, in its raw form, was not readily applicable to event classification, mostly due to three reasons: (1) the data format of the original PMU measurements, (2) the high data dimensionality, and (3) the measurements' quality issues from Section II-B related to missing, duplicate as well as noisy measurements. To alleviate these issues, the measurements underwent four stages of preprocessing prior to event classification.

1) Measurement cleansing. In the first stage, a data cleansing algorithm iterated the entire dataset of measurements in order to filter out PMUs reporting noisy measurements with

unreasonable scales across known voltage levels. It used simple thresholds to determine if the data stream exists, i.e. if there are any missing measurements, and then it removed duplicate measurements. All of the aforementioned steps were taken as precautionary preprocessing steps given that the provided status bits (refer to Section II-B) were deemed unreliable.

2) Measurement segmentation. The cleansed PMU measurements were divided, i.e. segmented into segments of measurements taken over 1-minute time intervals. The choice of the segmentation window size is based on the resolution on which the events were *labeled* and the reasoning behind it is discussed in Section III-C. Due to the different rates (30 fps and 60 fps) used to report the PMU measurements, a segment of 1-minute corresponds to 1,800 and 3,600 reported values when reporting with 30 fps and 60 fps, respectively. To avoid underutilization of data relevant to the event classification task, the latter reporting rates were down sampled through data averaging since the average as a central statistic is sensitive to peaks and other fluctuations in the measurements indicative of event occurrences. It should be noted that the only purpose of the down sampling is to get all PMUs' measurements to be consistent in terms of length (i.e. dimension) in order to represent a *valid input* to *data-driven models* aimed at learning to automatically detect and classify events by analyzing the measurements. Thus, down sampling of the original reporting rates of the respective PMUs did not affect the model robustness. This was empirically observed by considering only 60 fps measurements and comparing the event classification performance of the best-performing model in this study (discussed later in Sections IV-C and IV-D) *before* and *after* down sampling. The results are summarized in Table I.

TABLE I
EVENT CLASSIFICATION PERFORMANCE (IN TERMS OF AUPRC) ON MEASUREMENTS REPORTED AT 60 FPS BEFORE AND AFTER DOWN SAMPLING.

Before down sampling	*After* down sampling
0.871	0.867

It can be observed that the obtained performances before and after down sampling differ in less than 1% and can be considered rather comparable. This suggests that the down sampling did not have an impact on the subsequent event classification performance, supporting the model's robustness to preprocessing decisions of this sort.

3) Segment aggregation. One of the main stumbling blocks to data-driven classification, particularly when using traditional machine learning models, is high data dimensionality. In this study such a challenge was posed due to the high dimensionality (i.e. length) of the segments processed in stage *2)*. In addition, some of the segments had missing values (voltage and current segments in particular) as well as duplicate values which have been observed in segments from about third of the PMUs. The segments from all PMUs were *aggregated* to a lower resolution by binning the 1,800 values of every segment into non-overlapping buckets of 10 consecutive values each. The buckets were then summarized by taking the range of their constituent values, resulting in an aggregated segment representation of length 180. Such aggregation is expected to preserve the peaks caused by measurement fluctuations in the segments, which are presumably one of the key characteristics most informative for detecting and classifying events.

4) Segment summarization. The segment aggregation stage can bypass missing and duplicate values, nonetheless, any noise present in the initial segments might have easily breached into the aggregated segments. Moreover, although aggregated, having to consider segments generated by all PMUs for event classification increases the complexity of the problem. This poses a challenge particularly for recognizing local events, which are typically registered by only a few PMUs. Therefore, a joint representation of all PMU segments is considered. More precisely, for each time interval, a weighting scheme was used to summarize the aggregated segments for that specific time interval into a single segment, called a *barycenter*. The barycenters were computed by minimizing a differentiable loss function based on Soft Dynamic Time Warping (Soft-DTW) [31], designed specifically to preserve the smoothed dynamic distances over all possible alignments between time series (i.e. segments in our case). When computing each barycenter, the PMUs were assigned *importance weights*, calculated as the normalized standard deviations of their respective segments, and updated dynamically for every time interval. The benefits of such segment summarization are three-fold. First, noise can be filtered indirectly by the soft aggregation of the segments. Second, by considering a weighting scheme, even in the case of a single PMU registering a local event, such PMU's measurements would gain a larger weight and hence the event would be reflected in the corresponding barycenter without being dominated by normal operation measured by the other PMUs. Finally, as a biproduct, the overall measurements' dimensionality is additionally reduced since barycenters summarize the information from all PMUs' segments, subsequently allowing for lightweight event classification.

C. Event Labeling

Events of interest. In addition to *normal* system operation, this study considers two major event types: *local* and *system-wide*. The former is localized once it occurs (e.g., line faults), while the latter has a system-wide manifestation (e.g., fundamental frequency events). These two types of disturbances differ in the sense that a local event is typically registered in measurements from only one or a few PMU devices, whereas a system-wide event is expected to be registered in measurements from multiple PMUs located throughout the system. Examples of such contrasting events are presented in Fig. 2: a) the top subfigure shows a dip in voltage measurements from only one PMU (depicted in purple color). It indicates an event occurrence around 11:26:16, which was localized as it did not spread across the system or otherwise it would have been registered by some of the other PMUs.; and b) the bottom subfigure illustrates a different (system-wide) scenario in which the voltage measurements from all PMUs start fluctuating around the same time (08:22:15), clearly as a result of a disturbance which manifests itself across the entire system.

***Original (raw)* event log.** The original dataset was provided along with a collection of 4854 event occurrence records, referred to as *event log*. Each record contains an event's start and end timestamps, category (*event type/label*), cause (*planned*

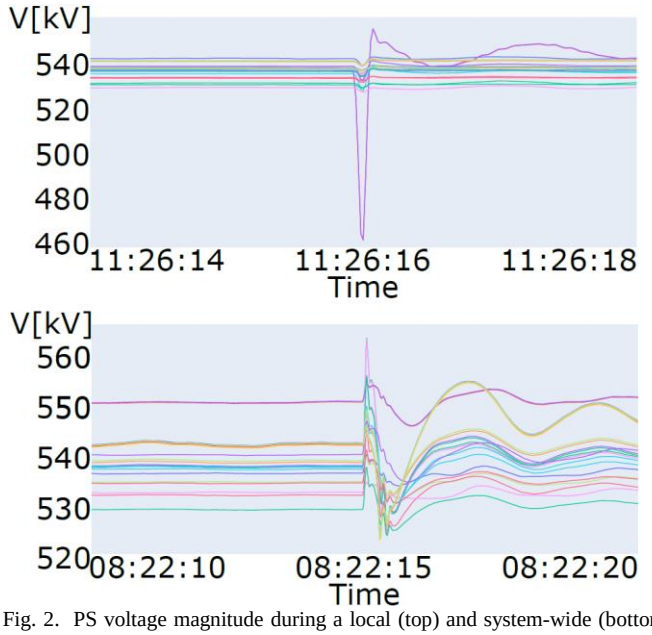

Fig. 2. PS voltage magnitude during a local (top) and system-wide (bottom) event. Measurements taken by different PMUs are depicted in different colors.

or *forced*), and a short description. Initially, the records of planned maintenance events were filtered out from the event log. Next, only records of line faults and fundamental frequency events, as event types representative of local and system-wide events, respectively, were considered.

The labels in the originally provided event log were created by different *operators* but not consistently applied since most likely the PMUs are located in substations of different operating companies. In addition, not all events that occurred in the period of 2 years (over which the measurements were collected) were initially labeled, while the labels for some events in the log were imprecise or unreliable, assigned probably without any verification involving a third party. Moreover, when creating the labels, the operators have not analyzed the PMU-level time-tag information to pinpoint their labels to particular timestamps in the PMU measurements. Instead, the time reference from SCADA and associated software tools was used, thus there is no exact correlation between the PMU timestamps and the time reference used by the operators. As a result, the time intervals in many event records were considerably broad and contained measurement segments not necessarily related to the event occurrences. Due to these data quality issues, the provided event log, in its raw form, was not directly applicable to the downstream task of event classification. To alleviate this shortcoming, we decided to involve a power system *domain expert* in reviewing the event labeling process to verify and refine the existing event records as well as to label additional events that the operators did not record in the original event log. This process was performed in three stages which resulted in the creation of three separate event logs, described as follows.

***Rapidly refined* event log.** The *original* event log was initially refined by the domain expert skimming thorough the log and removing the incorrect records in which the events were clearly mislabeled. The time intervals of the filtered records were generally quite broad and contained measurement segments around the events, considered to be less informative for event classification. Thus, the expert additionally narrowed down the time intervals to 1 minute each so as to eliminate some of the measurements unrelated to the events. The domain expert examined different interval lengths and selected the interval length of 1 minute based on the maximum duration of the observed events. Consequently, 1-minute intervals are expected to contain the entirety of the events (from the start to the end) without picking up considerable measurement segments around the events. Finally, 1-minute intervals of normal system operation were also handpicked by the domain expert and included in the refined event log.

***Partially inspected* event log.** Another version of the *original* event log was created in which the domain expert narrowed down the events' time intervals to a minute resolution, as with the *rapidly refined* log. As multiple events may have occurred within the same time interval, event records with such intervals were divided into several separate records. Subsequently, all records were adjusted so that the events are 'centered' approximately in the middle of the intervals. This 'standardization' of the events' positions within their respective intervals is performed to aid the downstream event classification when using traditional models, discussed later in Section IV. The updated event records were then visually verified by the domain expert, while any events the expert additionally identified in the process were also included in the log. In contrast to the *rapidly refined* event log, normal operation records were selected automatically by extracting 3-minute measurement segments before the start of each event.

***Fully inspected* event log.** The *partially inspected* log was further improved by replacing the automatically selected normal operation records with records carefully handpicked by the domain expert thought visual inspection. The records of line and frequency events from the *partially inspected* log were visually inspected one more time and several additional events were discovered in the process. After completing these steps, the resulting event log was considered to be *fully inspected*.

For a summary of the advantages and limitations of the described event logs, the reader is referred to Table II.

TABLE II
ADVANTAGES AND LIMITATIONS OF DIFFERENT EVENT LOGS. THE DOMAIN EXPERT WAS WORKING APPROXIMATELY 3 HOURS/DAY (*). THE '+' SYMBOL INDICATES ADDITIONAL (SECONDARY) VISUAL INSPECTION.

Event log	Handpicked normal operation segments	Narrower time intervals	Single event per interval	Precise intervals (centered events)	Visually inspected events	Labeling time*
Rapidly refined	✓	✓	✗	✗	✗	38 hours
Partially inspected	✗	✓	✓	✓	✓	~2 months (120 hours)
Fully inspected	✓	✓	✓	✓	✓+	2.5 months (150 hours)

D. Associating Measurement Segments with Event Labels

The preprocessed measurement segments (Section III-B) were labeled using the information available in the three event logs (Section III-C). This was achieved by intersecting the time intervals of the segments with those of the event logs' records. If a time interval of an event record was found to overlap with a segment's time interval then the segment was labeled with the corresponding event type (*line* or *frequency*) from the record. Otherwise, such a segment was labeled as *normal operation*. This labeling process was repeated for each of the three event logs, resulting in three different event classification scenarios.

IV. Data-Driven Event Classification

Hypothesis. The main hypothesis of our study is that *local* and *system-wide* events can be detected and classified based on reported sparse PMU measurements, without any additional information about the underlying measurement or power system.

Assumptions. The model, vintage, and user settings of the PMUs are not revealed due to the security and data confidentiality policies of the contributors (utility companies). It is known that the PMUs are sparsely located across the power system, however, their locations (substations) and the grid topology are unknown due to the aforementioned reasons. This also holds for the specifics of the communication subsystem and the locations and exact number of Phasor Data Concentrators (PDCs).

Objective. The objective is to develop a supervised *data-driven model* that can be *trained on historical data* that includes (1) historical PMU measurements, and (2) event labels (i.e. event indicators or annotations). More specifically, let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$ be a set of labeled historical measurement segments of a single PMU-measured quantity (e.g., PS voltage magnitude or PS current magnitude or frequency). A segment $\mathbf{x}_i = [x_i^1, \dots, x_i^T] \in \mathbb{R}^T$ measured over an interval $[t_1, t_T]$ is assumed to be preprocessed (following the procedure in Section III-B) and associated with an event label $y_i \in \{0, \dots, C\}$. In this study, the possible event class labels are $y_i \in \{c_n, c_l, c_f\} \subset \mathbb{Z}^{0+}$, where c_n, c_l, c_f are used to denote 'normal operation', 'line fault', 'fundamental frequency event', respectively.

Upon training a model based on the data in $\mathcal{D}$, the use of the trained model is to *automatically classify (i.e. predict)* the event label y of a previously unobserved measurement segment $\mathbf{x}$.

A. Traditional Classification Models

From a machine learning perspective, the event classification problem formulated above can be approached as a typical 3-class classification problem. Thus, several traditional multiclass models can be leveraged for event classification, each of which is briefly described as follows.

Decision Tree (DT) [32]. A decision tree's nodes are created by splitting $\mathcal{D}$'s feature space such that each of a node's children contains a subset of $\mathcal{D}$ in which as much segments as possible are labeled with the same event. Once the k-th node is created, it is used to determine an optimal split

$$\theta^* = \arg\min_{\theta} \mathcal{L}_{DT}(\mathcal{D}^k, \theta) \tag{1}$$

where $\mathcal{D}^k \subseteq \mathcal{D}$ is a subset of segments at node k, $\theta = (j, \tau)$ is a candidate split that splits a feature $j = 1, \dots, T$ based on the threshold $\tau \in \mathbb{R}$, and $\mathcal{L}_{DT}(\cdot)$ is a loss function measuring the class impurity of node k when split using θ. This procedure is recursively repeated for the child nodes of node k until $\mathcal{D}$ cannot be further partitioned or a desired tree depth is reached. Once build, the DT's split conditions are used to 'navigate' an unlabeled segment $\mathbf{x}$ through the tree until a terminal (leaf) node l is reached. The probability of a class c is then determined as

$$P(y = c|\mathbf{x}) = 1/|\mathcal{D}^l| \sum_{y_i \in \mathcal{D}^l} I(y_i = c) \tag{2}$$

Multinomial Logistic Regression (MLR) [33]. In the case of MLR, the probability of an event y given a segment $\mathbf{x}$ is defined as an output of a softmax function

$$P(y = c|\mathbf{x}) = \frac{\exp(\mathbf{w}_c^\top \mathbf{x} + b_c)}{\sum_j \exp(\mathbf{w}_j^\top \mathbf{x} + b_j)} \tag{3}$$

where $\mathbf{w}_c \in \mathbb{R}^T$ and $b_c \in \mathbb{R}$ are the model's coefficients and bias term for class $c \in \{0, \dots, C\}$, respectively. Using (3), an MLR is fitted by minimizing the categorical cross-entropy loss:

$$\mathcal{L}_{MLR} = -\sum_{c=0}^{C} \sum_{i=1}^{N} I(y_i = c) \log P(y_i = c|\mathbf{x}_i) + I(y_i \neq c) \log\big(1 - P(y_i = c|\mathbf{x}_i)\big) \tag{4}$$

The above loss is minimized using gradient-based methods since $\mathcal{L}_{MLR}$ is convex and its optimization is unconstrained.

Feed-Forward Neural Network (FFNN) [34]. A conventional, single-layer, neural network would map each segment $\mathbf{x}_i$ to a hidden, lower dimensional, feature vector

$$h(\mathbf{x}) = \mathbf{W}^\top \mathbf{x} + \mathbf{b} \tag{5}$$

where $\mathbf{W} \in \mathbb{R}^{T \times H}$ and $\mathbf{b} \in \mathbb{R}^H$ denote the projection matrix and the corresponding bias vector, respectively. Thereafter, the model parameters are learned by minimizing the categorical cross-entropy loss w.r.t. the hidden vectors:

$$\mathcal{L}_{FFNN} = -\sum_{c=0}^{C} \sum_{i=1}^{N} I(y_i = c) \log P\big(y_i = c\big|h(\mathbf{x}_i)\big) + I(y_i \neq c) \log\Big(1 - P\big(y_i = c\big|h(\mathbf{x}_i)\big)\Big) \tag{6}$$

Multiclass Support Vector Machine (MCSVM) [35]. A *binary* classification SVM learns the coefficients of a hyperplane that optimally separates *two* classes of data. The dual formulation of this objective is given by

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^\top \mathbf{Q} \boldsymbol{\alpha} - \mathbf{e}^\top \boldsymbol{\alpha} \tag{7}$$

$$\text{subject to } [y_1, \dots, y_N]^\top \boldsymbol{\alpha} = 0 \qquad 0 \leq \alpha_i \leq \lambda, \forall i = 1, \dots, N$$

where $\boldsymbol{\alpha}$ are the dual coefficients, λ is a regularization parameter, $\mathbf{e} \in \{1\}^N$, and $\mathbf{Q} = \big[y_i y_j \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)\big]_{N \times N}$ with $\mathcal{K}\big(\mathbf{x}_i, \mathbf{x}_j\big)$ being a kernel describing the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$. After solving (7), the decision value for a segment $\mathbf{x}$ is calculated as $f(\mathbf{x}) = \sum_i y_i \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) + b$. The event class of $\mathbf{x}$ is determined by $\arg\min_c f_c(\mathbf{x})$, where f_c is an SVM separating all segments from class c and those from any class other than c.

B. Single-Channel Convolutional Neural Networks

Unlike traditional classification models, convolutional neural networks [36]-[37] include layers capable of performing convolution operations. This makes them particularly desirable

for modeling measurement changes in response to external events. Since the problem formulation (beginning of Section IV) assumes $\mathbf{x}$ to be a univariate measurement segment (e.g., a segment of PS voltage magnitude or PS current magnitude or frequency), a so-called *single-channel* CNN (SC-CNN) can be leveraged. In this context, 'single-channel' refers to SC-CNN's capability to process homogeneous measurements of a single PMU-measured quantity. The building blocks of SC-CNN are described as follows.

Convolutional layer. The central component of an SC-CNN is the convolutional layer which transforms an input segment $\mathbf{x}$ by convolving it with a ω-sized filter (or kernel) $\mathbf{k}_k$:

$$v_{ik} = \mathbf{x}_{i:i+\omega}{}^{\top}\mathbf{k}_k \qquad \forall i = 1, \dots, T-\omega+1 \tag{8}$$

where $\mathbf{x}_{i:i+\omega} = [x_i, \dots, x_{i+\omega-1}]$. The operation (8) is computed for each $k = 1, \dots, K$, meaning that K different filters are applied to $\mathbf{x}$, whose outputs are passed through a rectified linear activation function, i.e. $\hat{v}_{ik} = \max(0, v_{ik})$. Note that the filters are not user-defined, but are automatically learned instead.

The activated filter outputs $\hat{\mathbf{v}}_k = [\hat{v}_{ik}]_{i=1}^{T-\omega+1}$ can be thought of as different transformations of $\mathbf{x}$, from which features are to be further learned automatically. Nonetheless, $\hat{\mathbf{v}}_1, \dots, \hat{\mathbf{v}}_K$ are expected to preserve the discriminative shapes regardless of where they appear in $\mathbf{x}$, which allows for learning time-invariant warped features in the subsequent layers [37]. One can define an arbitrary number of convolutional layers; the more layers are included, the "deeper" the network is considered to be. Depending on the learning problem, more layers might need to be employed to learn non-linear features of higher-order that capture certain discriminative characteristics which otherwise would not be perceptible after a single convolution.

Pooling layer. To emphasize the high-response characteristics in the convolved segment, the features produced by each filter are summarized through global average pooling:

$$g_k = \frac{1}{T-\omega+1}\sum_{i=1}^{T-\omega+1} \hat{v}_{ik} \tag{9}$$

The pooling operation has an effect of down sampling the filtered (convolved) features, while being robust to changes in the positions of the shapes preserved in those features. It is also crucial as it introduces a certain degree of *scale invariance* (i.e. *resolution invariance*). The features extracted upon pooling the measurements are, to some extent, resilient to scale changes (which in this study are posed by the two different measurement reporting rates).

Hidden and output layers. The pooled features $\mathbf{g} = [g_1, \dots, g_K]$ are mapped to a hidden representation:

$$h(\mathbf{g}) = \mathbf{W}^{\top}\mathbf{g} + \mathbf{b} \tag{10}$$

Subsequently, the softmax function (3) is used to estimate the event class probabilities $P(y = c|h(\mathbf{g})), \forall c \in \{0, \dots, C\}$.

Parameter learning. The parameters from all layers of the SC-CNN, $(\mathbf{k}_k^*, [\mathbf{W}^*, \mathbf{b}^*], [\mathbf{w}_c^*, b_c^*])$, are learned by minimizing

$$\mathcal{L}_{SCCNN} = -\sum_{c=0}^{C}\sum_{i=1}^{N} I(y_i = c)\log P\big(y_i = c\big|h(\mathbf{g}_i)\big) + I(y_i \neq c)\log\Big(1 - P\big(y_i = c\big|h(\mathbf{g}_i)\big)\Big) \tag{11}$$

C. Multi-Channel Convolutional Neural Networks

Multi-channel CNNs (MC-CNNs) are a generalization of SC-CNNs, capable of leveraging multivariate PMU measurements. Namely, abnormal system behavior may not always be perceptible from the measurements of a single PMU-measured quantity. Moreover, patterns that further characterize different events typically occur across measurements of multiple PMU quantities. To capture such patters relevant to event classification by utilizing all available PMU quantities, two multi-channel CNN architectures are presented.

Parallel Channel Filtering CNN (PCF-CNN). Following the formulation (beginning of Section IV), $\mathbf{x}$ is assumed to be a segment of measurements of a single PMU quantity. Thus, $\mathbf{x}$ can carry information only about the PS voltage magnitude or PS current magnitude or system frequency, measured by all PMUs in the system. In contrast to SC-CNN or any traditional model, a PCF-CNN utilizes the segments of all available PMU quantities. In that regard, let $\mathbf{x}^{(m)}$ be a segment of measurements of the m-th PMU quantity, for all $m = 1, \dots M$. An M-channel PCF-CNN filters the segments of M quantities separately, thus performing *parallel* channel filtering:

$$\hat{v}_{ik}^{(m)} = \max\Big(0, \mathbf{x}_{i:i+\omega}^{(m)}{}^{\top}\mathbf{k}_k^{(m)}\Big) \tag{12}$$

Upon filtering, the features produced by each channel are separately summarized through global average pooling and passed to a hidden layer as

$$h\big(\mathbf{g}^{(1)}||\,\mathbf{g}^{(2)}||\,\dots||\,\mathbf{g}^{(M)}\big) \tag{13}$$

where $\mathbf{g}^{(m)}$ are the features pooled from the m-th channel and $||$ is a concatenation operator. The event class probabilities are then estimated using (3) and the parameters from all layers are determined by minimizing (11).

Simultaneous Channel Filtering CNN (SCF-CNN). Unlike PCF-CNN, this architecture aims to leverage the segments of all PMU quantities by filtering all channels simultaneously. Formally, the key difference to PCF-CNN is that SCF-CNN learns a joint set of filters $\{\mathbf{k}_1, \dots, \mathbf{k}_K\}$ for all channels, i.e.

$$\hat{v}_{ik}^{(m)} = \max\Big(0, \mathbf{x}_{i:i+\omega}^{(m)}{}^{\top}\mathbf{k}_k\Big) \tag{14}$$

The activated filter outputs from each channel are then aggregated and subsequently pooled, i.e. $\mathbf{g} = \sum_m \hat{\mathbf{v}}_k^{(m)}$. The rest of the architecture is identical to that of PCF-CNN.

D. Model Variants

Standard variants. According to their definitions, the models described in Sections IV-A, IV-B, and IV-C are addressing the event classification problem by directly classifying segments into one of $C+1$ classes (including the normal operation class); thus, they are considered *standard* model variants. In that regard, upon obtaining the optimal parameters $\mathbf{\Theta}^*$, a standard classification model predicts the event class of a segment $\mathbf{x}$ as the most probable class given $\mathbf{\Theta}^*$:

$$f(\mathbf{x}; \mathbf{\Theta}^*) = \arg\max_c P(y_i = c|\mathbf{x}; \mathbf{\Theta}^*) \tag{15}$$

Hierarchical variants. Considering the nature of the event classification problem, it is unique in the sense that not all classes differ in the same manner. More precisely, from the domain it is known upfront that the normal operation class is expected to be much more distinctive as it is the only class carrying information about normal system operation, while all other classes represent different types of system disturbances.

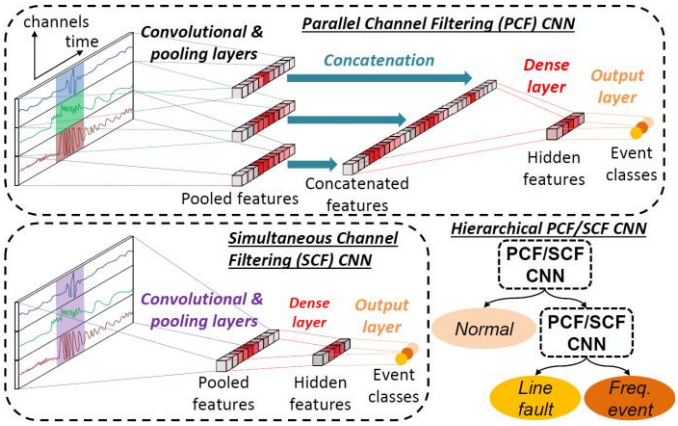

Fig. 3. Model architectures of standard and hierarchical PCF/SCF-CNN.

Therefore, *hierarchical* variants of event classification models can be leveraged to account for the distinctiveness of the normal operation class. Given that the considered event classes in this study are c_n, c_l and c_f, bi-level hierarchical model variants can be defined to break down the 3-class problem into two binary classification problems which are less challenging to address. In that regard, a hierarchical model would essentially represent a cascade of two consecutive binary classifiers; the first trained to distinguish normal (c_n) from abnormal ($c \neq c_n$) segments, followed by another classifier tailored to further classify the detected abnormal events into line faults (c_l) or frequency events (c_f). In essence, the former has the role of an *event detector*, while the latter can be considered an *event recognizer*. Now, given a segment $\mathbf{x}$, let us denote the raw outputs (*decision values* or *logits*) of the former and latter classifiers by $f(\mathbf{x})$ and $g(\mathbf{x})$, respectively. The event detector models the event probability as a sigmoid function $P_f(y = c_n|\mathbf{x}) = \sigma\big(f(\mathbf{x})\big) = 1/(1 - e^{-f(\mathbf{x})})$, also denoted by p for simplicity. Similarly, $q = P_g\big(y = c_f|\mathbf{x}\big) = \sigma\big(g(\mathbf{x})\big)$ is estimated by the event recognizer, with c_f being the 'positive' class. The hierarchical class probabilities are then calculated as

$$\big[P(c_n|\mathbf{x}), P(c_l|\mathbf{x}), P\big(c_f|\mathbf{x}\big)\big] = \begin{cases} [p, p/2\,, p/2\,], & \text{if } p \leq 0.5 \\ [0, 1-q, q], & \text{otherwise} \end{cases}$$

For an illustration of the standard and hierarchical variants of the multi-channel CNN models, the reader is referred to Fig. 3.

V. Experimental Results

A. Experimental Setup

Data preparation. The preprocessed version of the two-year PMU measurement dataset (Section III) was used in all conducted experiments. To account for patterns characterizing the different seasons throughout the year, the dataset was initially split into a *training* set containing all measurement segments recorded in 2016, and a separate *holdout* (test) set that encompassed all measurement segments from 2017. The training segments were labeled using the event logs described in Section III-C, thus producing *three labeled versions* of the training set. The segments in the holdout set were carefully labeled and thoroughly verified by the domain expert through visual inspection, to allow for representative assessment of event classification performance. For the distribution of segments among the event classes in the training set (under each labeling scenario) and the holdout set, refer to Table III.

TABLE III
Distribution of segments labeled as normal operation (NO), line fault (LF), or fundamental frequency (FF).

Event class	**Training** using different event logs('16)			**Testing**('17)
	Rapidly refined	*Partially inspected*	*Fully inspected*	Holdout
NO	467	1311	481	426
LF	454	227	229	273
FF	249	210	211	180
Total	1170	1748	921	879

Model parameter settings. Throughout all experiments, the traditional data models were run with the following settings: DT used Gini Index as a class impurity measure without a tree depth limit; FFNN utilized one hidden layer with a hidden dimension of $h(\mathbf{x}) = 30$; MCSVM was run with a linear kernel and a regularization penalty of $\lambda = 1$. Each of the CNN-based models (SC-CNN, PCF-CNN, and SCF-CNN) incorporated two convolutional layers, followed by one hidden layer. The initial convolutional layer utilized $K = 150$ filters, each of size $\omega = 30$, with 100, 15-sized filters being employed in the subsequent layer. In consistence with FFNN, the hidden layer dimension was set to 30. All neural network-based models were trained in 30 epochs with a batch size of 16, and a learning rate of 0.001.

Computational infrastructure. The raw measurements (Section III-A) were queried and preprocessed using Apache Spark 3.0 on an Apache Hadoop HPC cluster. All data models were implemented in Python 3.7 and TensorFlow 1.15 and run on an Ubuntu Linux 18.04.5 machine with 64 GB of memory and a 40-core Intel(R) Xeon(R) Gold 6230 CPU at 2.1 GHz.

Evaluation metrics. All models were evaluated using well-established classification metrics [38]-[39] including: (1) *AUPRC (Area Under the Precision-Recall Curve)*, *Precision*, *Recall* and *F1-score*. Due to the multiclass problem formulation the values of each metric were *macro*-averaged over all classes.

B. Overall Effect of Event Labeling on Event Classification

To inspect the effect that the event labeling process has on the downstream event classification performance, all traditional and CNN-based data models were compared using each of the three event logs (Table III), thus defining *three labeling scenarios*. Both the standard and hierarchical variants of each model were run. Due to space limitation only the performances of the best-performing hierarchical models are reported. The obtained results in terms of the evaluation metrics (Section V-A) are summarized in Table IV. Note that the names of the best-performing hierarchical models are prefixed with an "H".

One can observe that the CNN variants outperform traditional models. Specifically, the convolutional neural network models that consider the voltage measurements (either in a single or in a multi-channel mode) generally outperform the traditional classification models, as well as the single-channel CNNs trained on the current or frequency measurements. This suggests that voltage measurements are more informative for event classification compared to current and frequency. Multi-channel CNNs improve further upon SC-CNN (*V*) with HSCF-CNN being the best-performing model on the rapidly refined event log while HPCF-CNN is showing great advantage over

TABLE IV
EVENT CLASSIFICATION PERFORMANCE FOR DIFFERENT LABELING SCENARIOS.

Model variant \ Metrics		AUPRC	Precision	Recall	F1-score
Rapidly refined					
Traditional	DT	0.622	0.725	0.740	0.726
	MLR	0.802	0.783	0.758	0.754
	FFNN	0.845	0.778	0.769	0.758
	MCSVM	0.816	0.770	0.764	0.759
Single channel	SC-CNN (*V*)	0.803	0.803	0.797	0.800
	SC-CNN (*I*)	0.670	0.618	0.668	0.610
	SC-CNN (*f*)	0.650	0.598	0.647	0.592
Multi channel	PCF-CNN	0.804	0.833	0.831	0.818
	SCF-CNN	**0.854**	**0.841**	**0.845**	**0.836**
	HPCF-CNN	0.874	0.824	0.840	0.827
	HSCF-CNN	**0.897**	**0.856**	**0.861**	**0.857**
Partially inspected					
Traditional	DT	0.624	0.744	0.713	0.721
	MLR	0.848	0.820	0.765	0.773
	FFNN	0.824	0.833	0.749	0.769
	MCSVM	0.859	0.798	0.772	0.779
Single channel	SC-CNN (*V*)	0.867	0.863	0.803	0.824
	SC-CNN (*I*)	0.726	0.783	0.731	0.715
	SC-CNN (*f*)	0.700	0.769	0.687	0.695
Multi channel	PCF-CNN	**0.909**	**0.884**	**0.839**	**0.855**
	SCF-CNN	0.876	0.863	0.793	0.816
	HPCF-CNN	**0.915**	**0.898**	**0.854**	**0.871**
	HSCF-CNN	0.912	0.865	0.812	0.827
Fully inspected					
Traditional	DT	0.690	0.785	0.793	0.788
	MLR	0.856	0.821	0.807	0.808
	FFNN	0.838	0.836	0.830	0.832
	MCSVM	0.909	0.837	0.834	0.828
Single channel	SC-CNN (*V*)	0.906	0.871	0.865	0.861
	SC-CNN (*I*)	0.682	0.795	0.766	0.753
	SC-CNN (*f*)	0.696	0.774	0.714	0.731
Multi channel	PCF-CNN	**0.929**	**0.901**	**0.879**	**0.888**
	SCF-CNN	0.922	0.885	0.843	0.858
	HPCF-CNN	**0.940**	**0.911**	**0.891**	**0.900**
	HSCF-CNN	0.938	0.894	0.878	0.885

all models in the cases of the other two event logs. Another important observation is that there is a lift in classification performance as more curated event logs are used. This is due to (1) the manual handpicking of event-related segments by the domain expert, and (2) the additional time and effort dedicated to further handpick normal operation segments in the fully inspected event log. The latter leads to additional performance improvement since some of the events may span parts of their preceding 3-minute periods; which is the case with the partially inspected event log.

To further inspect how the performance of the best-performing multi-channel CNNs translates into specific numbers of detected events, their corresponding confusion matrices were visualized for each event log and presented in Fig. 4. First, the improvements introduced by the domain-based inspection of abnormal events are evident. Once abnormal events are partially inspected by the domain expert, additional 21-33 normal events are correctly classified and additional ~19 events are correctly detected as line faults. However, there is a considerable number of 16-25 misclassified frequency events which is most likely due to the labelling of the time period preceding each event as 'normal operation'. Also, recall that the partially inspected log required 3 times more effort for preparation compared to the initial, rapidly refined event log (refer to Table II). In case 25% more time is devoted to fully inspect normal operation measurements, additional 9-25 line and 9-14 frequency events are captured. Finally, this also leads

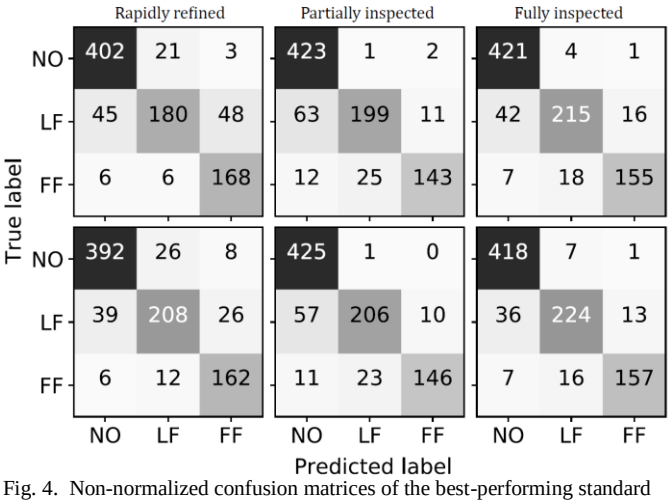

Fig. 4. Non-normalized confusion matrices of the best-performing standard (top row) and hierarchical (bottom row) multi-channel CNNs, under each event labeling scenario.

to an increase in detected line (and normal) events without compromising the detection of normal/frequency events.

C. Gradual Effect of Event Labeling on Event Classification

The outcomes of the experiment discussed in Section V-B revealed that HPCF-CNN, when trained using the fully inspected labels, outperforms the rest of the classification models, across all three labeling scenarios. The event labeling process is undoubtedly a time-consuming process and thus a question that arises is whether HPCF-CNN could have obtained similar performance had it been trained on a smaller fraction of fully inspected labels. Therefore, in the following experiment, the effect of event labeling on the classification performance of HPCF-CNN was analyzed in more detail. Initially, all segments were labeled using the rapidly refined event log. The rapidly refined labels were thereafter gradually replaced by fully inspected labels, adding one month of fully inspected labels at a time. For a given number of months n, the rapidly refined labels of events occurring during n randomly chosen months were replaced by their respective fully inspected labels. For each $n = 1, \ldots, 12$, this random replacement of labels was repeated 10 times. In each run, the performance of HPCF-CNN was assessed in case (1) the rapidly refined labels were replaced by fully inspected labels, and (2) if only the fully inspected labels were kept. The average testing F1-scores over all runs, along with their standard deviations, are presented in Fig. 5.

An initial observation from Fig. 5 is that the event classification performance clearly improves as larger amounts of fully inspected segments are considered. This holds in both cases when the fully inspected segments are used alone and in addition to the rapidly refined segments. In the latter case, four distinctive regions (depicted using different colors in Fig. 5) can be observed based on the largest performance gaps. In the first (purple) region, there is already a sudden lift in performance even when only 2 months of rapidly refined segments are replaced by fully inspected segments. After that the performance stabilizes up until 5 months of fully inspected data are replaced (red region in Fig. 5). This process takes from 15.7 to 28 days of inspection time i) depending on the number of months of data being inspected and ii) considering that the domain expert devoted approximately 3 hours per day for inspection (recall Table II).

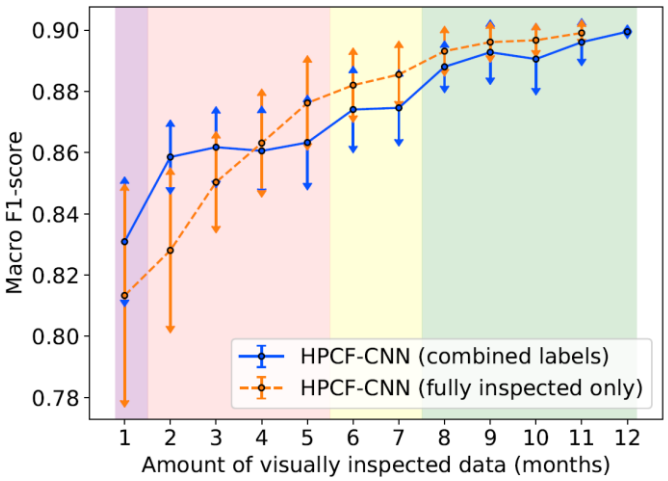

Fig. 5. Effect of the amount of fully inspected event labels on HPCF-CNN's event classification performance, when used alone (orange line) or in addition to the rapidly refined labels (blue line).

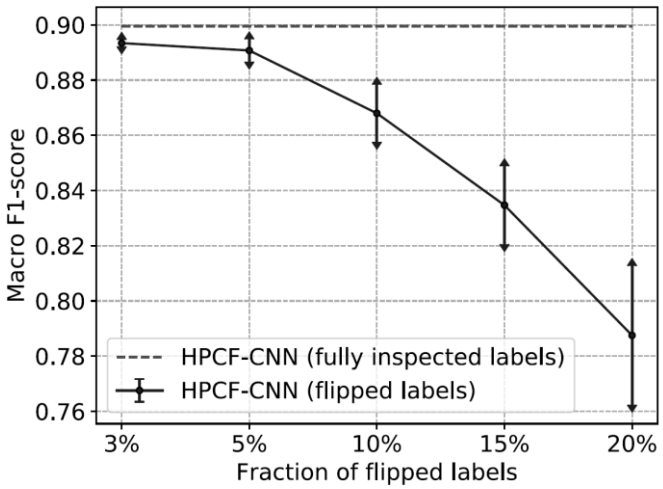

Fig. 6. Event classification performance for different fractions of flipped labels.

Subsequently, there is a lift in performance once 6 months of segments are replaced by expert-inspected segments and no significant change in performance was observed for 7 months (yellow region in Fig. 5). Thus, assuming that 5 months of data have already been fully inspected, achieving the performance lift to enter the yellow region requires an inspection of only one more month of data. This translates into 12% more inspection time relative to the time devoted to inspecting 5 months of data.

The last considerable gap in performance occurs when the coverage of fully inspected data is increased to 8 months and, from this point onward, HPCF-CNN's performance is considerably similar to that of the best-performing HPCF-CNN trained on all 12 months of fully inspected data (green region in Fig. 5). Fully inspecting the events for such a time period of data required the expert to devote between 2 and 2.5 months; i.e. from ~20% up to ~60% more inspection time than the time needed to achieve the yellow-region performance (with 6 months of fully inspected data).

In summary, if the domain expert's time is extremely limited, fully inspecting at least two months of data is suggested. In case of much greater expert's availability, it is suggested that the domain expert devotes at least ~2 months for inspection which translates to 8 months of fully inspected data. In such a case, performance similar to that of the best-performing HPCF-CNN (trained on all 12 months of fully inspected data) may be achieved. A tradeoff between inspection time and performance would be to consider $\geq$ 6 months of data, requiring around 6 weeks of domain expert's time to inspect. Finally, if more than 4 months of data are considered for inspection, using solely the fully inspected segments to train HPCF-CNN appears to be a consistently better choice than using them in addition to rapidly refined segments for the remaining months. This indicates that once 4 months of data are inspected, the quality of the fully inspected labels is already evident. On the contrary, the rapidly refined labels seem to be unreliable given that they lead to performance deterioration once used in addition to the fully inspected labels.

D. Domain Expertise Assessment

It is clear that the observations on HPCF-CNN's performance made in the afore discussed experiments hold assuming a certain satisfactory domain expertise. Hence, it is of considerable importance to investigate to what extent such observations would hold in case a less-experienced domain expert inspects the data. To quantitatively inspect the impact that a supposedly less experienced domain expert would have, lower degrees of expertise were simulated by considering different fractions of segments and randomly *flipping* their otherwise fully inspected labels. For each considered fraction, the experiment was repeated 10 times. The resulting performance of HPCF-CNN is presented in Fig. 6. First, it can be observed that the classification performance clearly drops with the increase of the flipping fraction. Larger fractions also enforce more 'randomness' in selecting which labels will be flipped, thus leading to larger performance fluctuations (reflected in the standard deviations of the average F1-scores). Even when 3% or 5% of the labels are flipped, the average F1-scores are quite similar to that of the HPCF-CNN utilizing the fully inspected labels. Significant drops in performance occur after flipping more than 5% of the labels. This observation suggests that similar performance may be achieved with the (supposed) expert as long as the expert is experienced enough not to mislabel more than 5% of the data.

E. Case Study: Event Detection

Considering that HPCF-CNN was observed as the model obtaining the best overall classification performance, its event detection performance was also inspected for each of the three event classes individually. The classification metrics were measured in a one-vs-rest manner and presented in Fig. 7. It can be observed that the precision is nearly consistent across all three classes which makes HPCF-CNN also a stable event detector although in this study it was leveraged for event

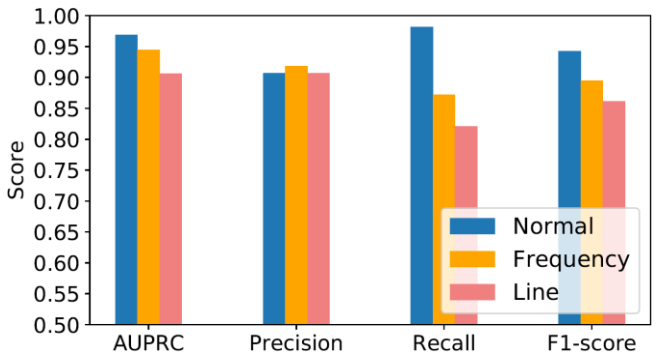

Fig. 7. Classification performance of HPCF-CNN w.r.t. each class individually.

classification. The largest lifts, on the other hand, are obtained w.r.t. recall. The normal operation segments can be easily detected, followed by fundamental frequency events which seem less challenging to detect than line faults. Intuitively, this is most likely due to the system-wide nature of frequency events as opposed to the locality of line faults.

VI. Conclusion

This study leverages several CNN model variants for detecting and further characterizing local and system-wide events. To assess their effectiveness, extensive experiments were conducted on two years' worth of PMU measurements from the Western Interconnection of the U.S. power grid, under three different *domain-expert-assisted labeling scenarios*. The main findings are summarized as follows:

- CNNs outperform traditional classification models regardless of the quality of the available event labels. CNNs utilizing multiple PMU-measured quantities improve further upon single-quantity alternatives. Multi-channel hierarchical CNNs outperform all alternatives considered.
- HCNNs' classification performances gradually improve as more measurement data is being inspected by a domain expert. Curating event logs leads to an increase in detected line faults, while maintaining a similar number of detected frequency events.
- If the domain expert's time is extremely limited, labeling at least 2 months of data is suggested. In case of greater availability, expert-inspected labels for at least ~8 months are needed to achieve satisfactory performance.
- In most cases, using smaller fractions of expert-inspected labels alone yields greater event classification performance than using them in addition to labels that were not fully inspected by a domain expert.
- Performance similar to that of the best-performing HCNN (based on entirely domain-driven labeling) may be achieved with a less experienced event curator as long as < 5% of event labels (used for model training) are mislabeled.
- Finally, normal operation segments were least challenging to detect, followed by frequency events which seem to be easier to detect than line faults, thus reflecting their system-wide nature as opposed to the locality of line faults.

VII. Disclaimer

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

VIII. References

[1] North American SynchroPhasor Initiative, "Data mining techniques and tools for synchrophasor data," NASPI White Paper, NASPI-2018-TT-007, January 2019.

[2] M. Biswal, Y. Hao, P. Chen, S. Brahma, H. Cao, and P. De Leon, "Signal features for classification of power system disturbances using PMU data," in *2016 Power Systems Computation Conference*. IEEE, 2016, pp. 1–7.

[3] S. Brahma, et al., "Real-time identification of dynamic events in power systems using PMU data, and potential applications—models, promises, and challenges," *IEEE transactions on Power Delivery*, vol. 32, no. 1, pp. 294–301, 2016.

[4] S. Wang, P. Dehghanian, and L. Li, "Power grid online surveillance through PMU-embedded convolutional neural networks," *IEEE Transactions on Industry Applications*, vol. 56, no. 2, pp. 1146–1155, 2019.

[5] M. Biswal, S. M. Brahma, and H. Cao, "Supervisory protection and automated event diagnosis using PMU data," *IEEE Transactions on power delivery*, vol. 31, no. 4, pp. 1855–1863, 2016.

[6] L. Xie, Y. Chen, and P. Kumar, "Dimensionality reduction of synchrophasor data for early event detection: Linearized analysis," *IEEE Transactions on Power Systems*, vol. 29, no. 6, pp. 2784–2794, 2014.

[7] O. P. Dahal, S. M. Brahma, and H. Cao, "Comprehensive clustering of disturbance events recorded by phasor measurement units," *IEEE Transactions on Power Delivery*, vol. 29, no. 3, pp. 1390–1397, 2013.

[8] M. El Chamie, et al., "Physics-based features for anomaly detection in power grids with micro- PMUs," in *2018 IEEE International Conference on Communications*. IEEE, 2018, pp. 1–7.

[9] M. Khan, P. M. Ashton, M. Li, G. A. Taylor, I. Pisica, and J. Liu, "Parallel detrended fluctuation analysis for fast event detection on massive PMU data," *IEEE Transactions on Smart Grid*, vol. 6, no. 1, pp. 360–368, 2014.

[10] A. Shahsavari, M. Farajollahi, E. M. Stewart, E. Cortez, and H. Mohsenian-Rad, "Situational awareness in distribution grid using micro-PMU data: A machine learning approach," *IEEE Transactions on Smart Grid*, vol. 10, no. 6, pp. 6167–6177, 2019.

[11] M. Kesici, C. B. Saner, M. Mahdi, Y. Yaslan, and V. I. Genc, "Wide area measurement based online monitoring and event detection using convolutional neural networks," in *2019 7th International Istanbul Smart Grids and Cities Congress and Fair (ICSG)*. IEEE, 2019, pp. 223–227.

[12] W. Li, D. Deka, M. Chertkov, and M. Wang, "Real-time faulted line localization and PMU placement in power systems through convolutional neural networks," *IEEE Transactions on Power Systems*, vol. 34, no. 6, pp. 4640–4651, 2019.

[13] J. Shi, B. Foggo, and N. Yu, "Power system event identification based on deep neural network with information loading" *IEEE Transactions on Power Systems*, vol. 14, no. 8, 2015.

[14] G. Xu, M. Liu, Z. Jiang, W. Shen, and C. Huang, "Online fault diagnosis method based on transfer convolutional neural networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 2, pp. 509–520, 2020.

[15] X. Ding and Q. He, "Energy-fluctuated multiscale feature learning with deep convnet for intelligent spindle bearing fault diagnosis," *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 8, pp. 1926–1935, 2017.

[16] L. Wen, X. Li, and L. Gao, "A new two-level hierarchical diagnosis network based on convolutional neural network," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 2, pp. 330–338, 2020.

[17] F. Ciancetta, et al., "A new convolutional neural network-based system for NILM applications," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–12, 2021.

[18] J. C. Mayo-Maldonado, et al., "Data-driven framework to model identification, event detection, and topology change location using D-PMUs," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 9, pp. 6921–6933, 2020.

[19] W. Wang, H. Yin, C. Chen, A. Till, W. Yao, X. Deng, and Y. Liu, "Frequency disturbance event detection based on synchrophasors and deep learning," *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 3593–3605, 2020.

[20] M. Jamei et al., "Anomaly Detection Using Optimally Placed μPMU Sensors in Distribution Grids," *IEEE Transactions on Power Systems*, vol. 33, no. 4, pp. 3611–3623, July 2018.

[21] W. Yu, W. Yao, Y. Zhao, and Y. Liu, "Timestamp Error Detection and Estimation for PMU Data based on Linear Correlation between Relative

Phase Angle and Frequency," in *Proceedings of the 53rd Hawaii International Conference on System Sciences*, January 2020.
[22] W. Yu, W. Yao, X. Deng, Y. Zhao, and Y. Liu, "Timestamp Shift Detection for Synchrophasor Data Based on Similarity Analysis Between Relative Phase Angle and Frequency," *IEEE Transactions on Power Delivery*, vol. 35, no. 3, pp. 1588–1591, June 2020.
[23] I. Niazazari and H. Livani, "Disruptive event classification using PMU data in distribution networks," in *2017 IEEE Power & Energy Society General Meeting*. IEEE, 2017, pp. 1–5.
[24] Y. Zhou, et al., "Abnormal event detection with high resolution micro-PMU data," *Power Systems Computation Conference*. 2016, pp. 1–7.
[25] Niazazari and H. Livani, "A PMU-data-driven disruptive event classification in distribution systems," *Electric Power Systems Research*, vol. 157, pp. 251–260, 2018.
[26] *IEEE/IEC International Standard - Measuring relays and protection equipment - Part 118-1: Synchrophasor for power systems - Measurements*, in IEC/IEEE 60255-118-1:2018, vol., no., pp.1-78, 19 Dec. 2018, doi: 10.1109/IEEESTD.2018.8577045.
[27] M. Kezunovic, S. Meliopoulos, V. Venkatasubramanian, and V. Vittal, *Application of time-synchronized measurements in power system transmission networks*. Chapter 1, pp 6-8, Springer, 2014.
[28] X. Liang, S. A. Wallace, and X. Zhao, "A technique for detecting wide-area single-line-to-ground faults," in *2014 IEEE Conference on Technologies for Sustainability (SusTech)*. IEEE, 2014, pp. 121–124.
[29] D. Nguyen, R. Barella, S. A. Wallace, X. Zhao, and X. Liang, "Smart grid line event classification using supervised learning over PMU data streams," in *2015 Sixth International Green and Sustainable Computing Conference (IGSC)*. IEEE, 2015, pp. 1–8.
[30] T. Dokic, M. Pavlovski, D. Gligorijevic, M. Kezunovic, and Z. Obradovic, "Spatially aware ensemble-based learning to predict weather-related outages in transmission," in *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019.
[31] M. Cuturi and M. Blondel, "Soft-DTW: a differentiable loss function for time-series," in *International Conference on Machine Learning*. PMLR, 2017, pp. 894–903.
[32] L. Breiman, et al., *Classification and regression trees*. CRC press, 1984.
[33] K. P. Murphy, *"Logistic regression," Machine learning: a probabilistic perspective*. Chapter 8, pp. 245–279, MIT press, 2012.
[34] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.
[35] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," *The Journal of Machine Learning Research*, vol. 5, pp. 101–141, 2004.
[36] Goodfellow, Y. Bengio, and A. Courville, *"Convolutional networks," Deep learning*. Chapter 9, pp 326–366, MIT press, 2016.
[37] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 917–963, 2019.
[38] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in *International Conference on Machine learning*. PMLR, 2006, pp. 233–240.
[39] N. Chinchor and B. M. Sundheim, "MUC-5 evaluation metrics," *Fifth Message Understanding Conference (MUC-5)*. August 25-27, 1993.

IX. Biographies

Martin Pavlovski (S'21) received the B.Sc. degree in Electrical Engineering and Information Technologies from the Saints Cyril and Methodius University of Skopje, Skopje, Macedonia, in 2015, and the Ph.D. degree in Computer and Information Science from Temple University, Philadelphia, PA, in 2021. Martin was a Researcher at the Macedonian Academy of Sciences and Arts, and a Visiting Scholar at Temple University. His research focuses on machine learning from structured data.

Mohammad Alqudah (S'20) received B.Sc. in Computer Engineering from Jordan University of Science and Technology, Irbid, Jordan, in 2012 and M.S. degree in Industrial and Systems Engineering from Binghamton University, NY, in 2015. Currently, he is a Ph.D. candidate in Computer and Information Sciences at Temple University. His main research interests are Machine Learning, Data Mining, and Big Data Analytics.

Tatjana Dokic (S'10-M'19) received M.Sc. and Ph.D. degrees in Electrical and Computer Engineering from the University of Novi Sad, Novi Sad, Serbia, in 2012, and Texas A&M University, College Station, TX, in 2019, respectively. Currently she is an Assistant Research Engineer at the Texas A&M University. Her main research interests are big data for power system applications, power system asset and outage management, weather impacts on power systems, time series analysis of PMU measurements, smart grids.

Ameen Abdel Hai (S'20) received B.Sc. in Software Engineering from the University of Northampton, U.K., in 2015, and M.S. degree in Computer Science from Saint Joseph's University, PA, in 2018. Currently, he is a Ph.D. candidate in Computer and Information Sciences, Center for Data Analytics and Biomedical Informatics at Temple University, PA. His main research interests are Machine Learning, Data Mining, and Big Data Analytics.

Mladen Kezunovic (S'77–M'80–SM'85–F'99–LF'17) received his B.Sc. from University of Sarajevo, Bosnia, and M.Sc. and Ph.D. in electrical engineering from University of Kansas, Lawrence, KS, in 1974, 1977, and 1980, respectively. He has been with Texas A&M University, College Station, TX, USA since 1986, where he is currently Regents Professor. For over 25 years he has been the Principal Consultant of XpertPower Associates specializing in power systems data analytics. His expertise is in protective relaying, automated power system disturbance analysis, computational intelligence, and smart grids. He has authored over 600 papers, given over 120 seminars, invited lectures, and short courses, and consulted for over 50 companies worldwide. Dr. Kezunovic is a CIGRE Fellow, Honorary and Distinguished member. He is Registered Professional Engineer in Texas.

Zoran Obradovic (S'85-M'87-SM'91) is a Distinguished Professor and a Center director at Temple University, an Academician at the Academia Europaea (the Academy of Europe) and a Foreign Academician at the Serbian Academy of Sciences and Arts. He mentored 45 postdoctoral fellows and Ph.D. students, many of whom have independent research careers at academic institutions and industrial research labs. Zoran is the editor-in-chief at the Big Data journal and the steering committee chair for the SIAM Data Mining conference. He is also an editorial board member at 13 journals and was the general chair, program chair, or track chair for 11 international conferences. His research results were published at about 400 data science and complex networks articles addressing challenges related to big, heterogeneous, spatial and temporal data analytics motivated by applications in healthcare management, power systems, earth and social sciences.