IoT Data Management System for Rapid Development of Machine Learning Models

Keith Grueneberg, Bongjun Ko, David Wood, Xiping Wang, Dean Steuer, Yeonsup Lim
IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
{kgruen, bko, dawood, xiping}@us.ibm.com, {dean.steuer, y.lim}@ibm.com

Abstract—Capturing and managing the data needed to build effective machine learning models for custom IoT environments requires a great deal of effort. The amount of data generated from IoT devices is abundant, but tools to find datasets appropriate for the desired models are lacking. This paper presents a data capture system and data management catalog with solutions addressing the challenges of curating IoT data applied to purpose-built machine learning deployments.

Keywords-Machine Learning, Internet of Things, Data Management

I. INTRODUCTON

The ever-growing number of connected devices in the Internet-of-Things (IoT) means an explosion of data generated by those devices ("things"). While the availability of the connectivity to and from the devices enables the remote monitoring and control from anywhere and at any time, the real value of IoT lies in the ability to consume, process, and analyze the data generated by the devices, and then to produce actionable insights about our surroundings and everyday objects of interests. Due to the speed, the volume, and the real-time nature of IoT data, however, such analytical capability can only be realized through intelligent agents running at scale in real-time, rather than by offline data analysis.

Artificial intelligence techniques are key to enabling such a scalable and real-time analytics solution to many IoT domain problems, and machine learning/deep learning (ML/DL) is becoming the most prominent approach to building the artificial intelligent agents for IoT data analytics. As with most other applications of ML techniques, the first step in ML-based IoT data analytics is to collect and curate a good amount of labeled data in order to train effective machine learning models. In some popular ML problems such as image analysis, speech recognition, and natural language processing, there are often large standard labeled datasets serving as benchmarks for performance evaluation as well as practical baselines for many real-world problems. However, when one attempts to solve IoT data analytics problems through machine learning techniques, there are additional challenges in obtaining the labeled data sets for IoT applications due to the following reasons:

- Many types of IoT data are simply time-series signals and are often difficult for non-experts of the domain to understand and interpret, which in turn makes it impractical to procure a large amount of labeled data through crowd-sourcing.
- IoT devices typically generate continuous streams of data 24x7, that could quickly produce a vast amount of unlabeled data. Finding "interesting" segments and assigning labels within that mass of data is a very laborious task.
- For many IoT applications, the moments of interests are often rare events, e.g., anomalous events within a long time-series signal are (by definition) rare. Therefore, even if they were effectively collected and labeled, it is not uncommon that the labeled dataset has unbalanced distribution across different labels in the data size.
- Publicly-available datasets (e.g. UCI Machine Learning Repository [1] and Kaggle [2]), provide little benefit in many IoT data analytics applications because IoT applications often have a very specific domain focus, e.g., identifying abnormal issues and their causes in a particular set and type of equipment on a manufacturing floor.

In this paper, we present an audio capture application and ML data catalog system for IoT analytics applications we built to overcome the above challenges. It is designed with the following principles:

1) Data acquisition for customized IoT environments must be made easy and integrate well with the data catalog to enable efficient curation of captured data.
2) It must be easy for users to browse through both the labeled data and unlabeled data, and then to assign, review, and modify data labels. This means the system needs to support the individually addressable data in its design of storage and APIs.
3) It must allow both the online labeling of the streaming data (i.e., in situ labeling) and the offline reviewing and label assignment with easy-to-use and intuitive user interface/experience (UI/UX)
4) It must minimize the labeling efforts by human users through a large volume of unlabeled IoT data, by suggesting or automatically applying labels.
5) It must allow the integration and combination of internally collected data and externally available data.

There is an extensive amount of previous work in this area including many commercial data catalog software products (e.g. Microsoft Azure [3] and Collibra [4]). While these products offer many advantages for enterprise data assets, the solution we present below is geared towards creating training sets for specific scenario from data collected from IoT sensors.

In the remainder of the paper, we will discuss use cases, our solutions and future work. As a canonical example, we will discuss our solutions in the context of acoustic analytics problem, that is, analyzing sound to understand the environments and status of various objects. Much of the discussion is equally applicable for other types of IoT data.

II. USE CASES

We present a real-world application of IoT analytics that we developed and used the sounds (non-speech) as the data

978-1-7281-2711-8/19/$31.00 ©2019 IEEE
DOI 10.1109/ICCC.2019.00021

source for detecting and understanding the status and the significant events of the objects and environments being monitored.

The first use case is the event and activity analysis of sporting events, such as tennis matches and golf tournaments [5], for which various sounds generated by the activities of players and audience are analyzed, e.g., sounds of the ball being hit (stroke or service in tennis, or driver, wood, and iron shots in golf), ball hitting the net, referees making the calls, players' footwork, audience sounds (clapping, cheering, or roaring). We use statistical methods and machine learning models to detect such moments of interests and analyze (classify) different types of activities and events, and then use the analysis results to (i) automatically collect match statistics, and/or (ii) (when accompanying video feed is available) extract the "interesting" and "significant" video clips of the matches (i.e., match highlights).

Another use case is the industrial sound analytics [6][7], in which the acoustic signals from various machines and devices (e.g., vehicles, manufacturing equipment, HVACs in buildings, etc.) are analyzed in real time, so that abnormal states of the equipment are detected, and the detected sounds are classified into different causes and types using machine learning models. Results of such analysis can then be fed into next-level analysis that determines the most appropriate actions to be taken, e.g., making a work order for a maintenance task for a specific part of the machine, generating an advice for the driver to take the vehicle to a repair shop, etc.

It is noteworthy that the above two categories of applications have both common and distinct challenges, in terms of data collection and labeling to be used to train the ML models. In both cases, the raw (unlabeled) dataset, when collected through continuous recording, contains a relatively small amount of "notable" moments, interleaved between long but not-so-interesting periods. Finding out those significant parts in a long stream (or recording) of sounds requires human labelers to review a long and large sound data, only to retrieve a small amount of labeled data, resulting in seriously low return-on-investment in terms of time and effort. The differences include the duration of the individual moments of interests (typically a short burst of sounds in sports events vs relatively persistent ones in industrial sounds), background noise profile (relatively quiet background noise in tennis/golf matches vs highly noisy environments in industrial setting), required level of expertise in labeling (significant domain expertise needed for industrial sounds vs less so in sports sounds), and the transferability of the labeled sets across different environments (relatively homogeneous across different sports event vs sensitive to the environments). We will elaborate these challenges in the next section.

III. CHALLENGES

The primary objective of our use cases is to have high accuracy acoustic recognition in local environments having unique acoustic signatures. To accomplish this, a customized model must be built for each environment or domain, which in turn requires building a robust training data set using sounds that represent the environment. The primary challenge we address in this paper is the development of the training data sets for these localized models. The remainder of paper will discuss how we addressed the challenges below.

Data Collection

The training sounds must be broad enough to cover all the sounds that need to be encountered and are to be recognized. It must also be as rich as possible, including the variations that might be encountered for each sound. Ideally, the training sounds are taken from the environment to be monitored.

Accurate Labeling

After capturing sounds, care must be taken to assure that the data set is as "clean" as possible. This means that the labeling of the data is atomic, accurate and balanced. Labeling is atomic when the label applies to the full length of the sound and not a portion of it. Of course, labeling needs to accurately represent ground truth. When possible, a data set that has an equal amount of training data for each of the labels is ideal. A more atomically labeled, accurate and balanced training set, will produce a more accurate model.

For machine learning algorithms to make accurate and distinct decisions about a dataset, the individual data points must also contain accurate and distinct labels. Multi-label data points introduce a further challenge in deciding how to classify data records as does the absence of labels entirely and the meaning of a label if the labels are being assigned by a human and therefore may not be strictly defined.

Scale

Another challenge is dealing with the massive amounts of data generated from IoT devices. A large volume of data is always preferred when training the acoustic model. However, curating and labeling the data is a laborious task and so we would like to provide automated assistance to the user applying labels and cleansing the data.

Unbalanced Training Data

Finally, unbalanced training data refers to the problem where machine learning datasets are not evenly distributed across the classes that the model is recognizing. This may cause inaccurate results in some classes which aren't represented equally as the other classes.

IV. SOLUTIONS

This section discusses how the challenges of creating a purpose-built dataset are addressed in our system. The system implements a data curation pipeline where the raw data collected is processed in *stages* from online collection and in-situ labeling, through offline reviewing and (re-)labeling, to data curation for ML model training. Note that this process can be typically performed in iteration, as the end-user (the ML modeler) can trigger processes in earlier stages depending on his/her needs and outcome from using the curated data. In

what follows, we describe this pipeline in detail (shown in Figure 1 below), including techniques we employ to facilitate the operations at each stage.

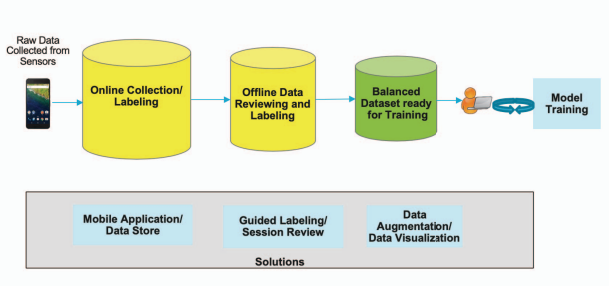

Figure 1 IoT Data Curation Pipeline

A. Data Store and Labels

One of the basic requirements in ML data management is the ability to efficiently address individual data samples, such that one can review their values, assign/re-assign their labels, and select for ML model training. Such an ability is critically needed because (i) it is not uncommon that the IoT data collected for specific domain problems have no or incorrect labels at the time of data collection, (ii) there are cases when only a subset of the dataset is used for model training, and (iii) a new dataset that combines portions of different dataset is used for model training. Also, it is desirable for a user to be able to access both external dataset (often collected and pre-labeled by others and stored in other data storage) and internally collected at the same time, in case one would like to use the combined dataset.

To maximally support different conventions used in data labeling and ML modeling, we support following types of data label for each data sample: (i) a single label, (ii) multiple (yet "flat") labels, (iii) hierarchically organized labels, i.e., label ontology, and (iv) multi-labels in key-value pairs, where the keys indicate the "type" of labels and the values indicate the actual labels to be used in ML models. In our experience, the last type of labels (key-value pairs) is particularly useful in industrial IoT scenarios, where not only the labels of the raw data but also other meta data, such as types, models, identity, location and other contexts about the objects being monitored, can be included in the modeling process in order to build different types of models according to different contexts. For example, in the automotive sound analysis application, a sound clip recorded in a car can be labeled as {category:car, model:Toyota-Camry-LE, source: wheel, state: abnormal, cause: bad-bearing, condition: high-speed}, and the user can choose to collate the data samples according to his/her own choice in order to build a custom model for a specific purpose (e.g., model for abnormal sounds in a high-speed driving condition).

B. Online Data collection and Labeling

IoT data is generally in the form of streaming time-series yet the "interesting" moments within the data stream worthy of getting labeled are typically "ephemeral", that is, when those moments are passed unnoticed, they become very difficult to recognize later through offline review of the recorded data. Therefore, it is best to capture the moments and assign the labels *as they happen*. For this, we use a mobile application that supports the continuous recording of the sounds, yet easy-to-use capturing of sound segments and subsequent labeling of them in one of the aforementioned label formats. Capturing of sound segments is done in two ways:

1. Manually by the user: when he/she notices ("hears") sounds of interesting events/symptoms and presses a button to capture that particular sound segment.
2. Automatically by a pre-trained sound analysis model: The app runs a pre-trained sound analysis model, and automatically captures sound segments that are detected to belong to some pre-defined classes of sounds (e.g, "abnormal" sounds determined by an anomaly detection model trained with a baseline "normal" sounds).

A captured sound segment is processed in the following way: it is first stored temporarily in mobile device's local storage, and then either (i) assigned a label by the user immediately online on mobile app, or (ii) marked for a later, offline review and label assignment. The locally stored segments (either assigned labels or marked for later review) are pushed to the data catalog storage upon the next uploading opportunity.

The above staged process, that is, (1) manual/automatic capturing, (2) temporarily storing locally, (3) online labeling or marking for offline review, and (4) pushing to the server, allows for robust and maximum coverage of a variety of situations that may occur during data collection and labeling stage of IoT data, depending on, e.g.,

- whether a user is present or not at the moment an interesting sound segment is encountered (via manual vs automatic capture),
- whether a user can and/or knowledgeable enough to assign labels in real time (via online labeling vs offline review), and
- whether a reliable network connection is available at the time of sound capturing (via locally storing the captured sounds and the "pushing" them to the server when the connection is available).

C. Offline Data Reviewing and Labeling

In the previous sub-section, we presented methods to help filter out unnecessary data in long recordings of IoT data through online capturing (automatically or manually) only those parts significant for assigning the labels. The next step in the data curation pipeline [8] is to clean the collected data so that the data left unlabeled or mis-labeled during the online collection process is corrected and also excess or noisy data is removed. The data cleansing step, which is performed offline, is particularly needed because

- the online capturing process may not be available in the first place (e.g., no human labeler or data capture model

is available), ending up collecting a large amount of unfiltered and unlabeled data, and
- there are (almost always) errors in the manual or automatic capturing/labeling process (e.g., human labeler missing the moments of interests or incorrectly labeling the captured data, an anomaly detection method generating many false positives).

Consider, for example, a driver records sounds inside an automobile for an extended period, e.g. 30 minutes, for automotive sound analysis application, in a driving condition. Due to safety reasons, the driver is practically prohibited from providing any sort of labels to the sound being recorded, so the entire 30 minutes of the sound recording shall need to be reviewed and labeled later. The offline review and labeling process is essentially done manually by human through a set of user interfaces that help the user navigate, select, review, and assign/re-assign the labels for the data collected and uploaded to the data catalog.

One critical aspect of the data reviewing framework is the notion of "recording session", which indicates a temporal grouping of the collected data resulted from an "episode" of data capturing. It turns out, from our experience in the real-world experiments, that such a temporal grouping typically corresponds to the semantic grouping of the collected data in IoT scenarios. It thus provides the user not only a simple reminder of when and how the data in a session is recorded, but also, more critically, a convenient and intuitive way to assign the labels across different but strongly related pieces of data within the same session.

Beyond this basic functionality, our data catalog system is designed to further maximize the efficiency of the reviewing and labeling efforts, helped by software tools that can make the process intuitive and (semi-)automated as much as possible, such as intuitive data visualization, auto-segmentation, and guided labeling, which we describe in detail in the next section.

V. HELPER METHODS FOR DATA LABELING

A. Data Visualization

One challenge of large volume data is understanding the data and exploring it in a natural way. When provided with a dataset and an ontology which defines the relationship between the labels of a data record, we can build a visual representation of this ontology.

With this visualization of the ontology, we can explore relationships in the ontology based on the labels associated with individual records. Such a hierarchical representation of data and its visualization also facilitate easy iteration of purpose-built ML model training process, as the user can visually and efficiently select specific branches and sub-categories of the dataset to be included in the training dataset.

It may not be the case that a dataset comes with a predefined ontology illustrating the relationship between labels. Hence it is beneficial to generate this relationship, or a user may want to explore how best to represent the relationship between labels of data records. This is particularly useful when multiple labels are given to individual data samples. More specifically, given a set of labels for each record, we can build an ontology of the labels according to a user-defined ordering of the labels that define which one should be considered to be at a higher level in hierarchy than the others.

Additionally, there are cases where data records may not contain label values for some particular fields. In such an example, the ontology should make this apparent because investigation may reveal relationships and behavior of the data. Additionally, it may be the case that records lack values for certain fields, visualization helps to make this case more apparent and possibly provide insight into why this may happen. For example, is a record missing a label or is it purposely left blank can be indicated on visualization. Data visualization also helps identify overall properties of the larger dataset, e.g., intuitively and immediately identifying the uneven distribution of data points.

B. Guided (Assisted) Labeling

One of the main requirements to train supervised learning models is to be able to access annotated data that can be used to train ML models. As shown before, users of the data catalog can easily access recorded sounds and annotate files either individually or by groups. Although having the option to annotate files individually or by group is useful for the end user, it might be very time consuming and error prone process. For instance, if the time duration of each file is 30 seconds, and the total amount of files recoded by one user in a day is 12, it will take about 6 hours to listen and label all files in just a single session. To help users label large amounts of data, the data catalog includes a guided labeling process, where the user only needs to annotate a small set of audio files in order to train a ML model and get 'suggestions' about the labels for unannotated sound recordings. Using the annotated labels, the guided labeling process trains a ML classifier that is iteratively updated and used by the guided labeling algorithm to suggest the labels for the remaining unannotated sound recordings selected by a user.

Although guided labeling helps the user annotate sounds by providing suggested labels along with a confidence value associated, it still requires the user to provide an initial set of annotated data. Hence, we use Active Learning (AL) [9] as a method to reduce the number of interactions and work required from a user with the role of data annotator. An AL algorithm actively asks users for annotations of data points (e.g., sound recordings) from a set of unannotated data. After a sample, or sound recording, have been selected by the algorithm and labeled by the user, the annotated sample is added to the pool of training data. There exist different query selection methods that can be used to select unannotated files (i.e., samples) as queries for the user. The goal is to reduce the number of queries for the user by selecting only a small set of queries that allow an underlying ML model to learn quickly the features of the underlying data.

Our guided labeling framework is built in such a way that one can plug in and experiment with different types of query selection methods easily. We plan to investigate other query selection strategies and methods that allow us to improve the performance of the AL process when having a small budget of queries to the user i.e., only a small portion of the data can be labeled, so that after a set of queries the remaining unlabeled data can be automatically labeled.

C. Data Augmentation

Machine learning algorithms requires a proportional amount of training data to appropriately train their parameters for optimal performance. However, in real world, it is hard to collect refined training data (e.g., text annotated with its sentiment and audio with its source type annotation). Data augmentation is a process to generate additional data from given well-refined one so that generated data have similar characteristics with different contents to base data, e.g., "he is sad since his dog passed away" from "she cries when her cat dies": both for "sad" sentiment. Typically, techniques to perform data augmentation are categorized as follows:

1. **Simple data manipulation**: This approach is to add/remove/change values in base data using internal and external information from base data, e.g., adding gaussian noise in numerical data, horizontal/vertical flip of images, and shuffling frames in audio data. This is the simplest way to obtain additional data, but for some data types, generated data can be too distorted to represent the characteristics of base data, e.g., dog sound shuffling can produce weird sound which no more seems animal sound.
2. **Generative model-based approaches**: This approach is based on latest methods of data generation using neural networks, such as GAN (Generative Adversarial Network) [10] and VAE (Variational Auto-Encoder) [11], which provide more formal and general framework for data augmentation. For example, GAN consists of two components: generator and discriminator. The generator produces fake data from noises to deceive the discriminator and the discriminator tries to identify whether input data is real or fake. By training two components with these conflicting purposes, GAN can automatically generate artificial data that reflect characteristics of real data.

The above two approaches have their pros and cons. The approaches by the generative models would provide a more general method than the data manipulation method, which would typically require the domain-specific knowledge about the characteristics of the data. On the other hand, the generative approaches require a large amount of data to train the generative models in the first place. Our system employs both techniques so that the users can choose the best method according to his/her situation (e.g., the amount of existing data, the domain knowledge) to synthesize and obtain additional data together with refined data managed by our platform. This feature will allow users to train ML models with enough data to achieve better performance.

VI. Conclusions/Lessons Learned

In the course of building models for the use-cases we discussed in this paper, it became clear that the quality of the labeled datasets is as important as the machine learning models themselves. When building models for specific use-cases, such as detecting specific sounds, using publicly available datasets may not be sufficient. We have built a system which allows us to rapidly curate data collected from IoT sensors from the local environment so that accurate models can be built for the purpose required.

ACKNOWLEDGEMENT

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

REFERENCES

[1] UCI Machine Learning Repository - http://archive.ics.uci.edu/ml/datasets.html

[2] Kaggle Datasets - https://www.kaggle.com/datasets

[3] Microsoft Azure - https://azure.microsoft.com/en-us/

[4] Collibra - https://www.collibra.com/

[5] Verma, D., Ko, B. J., Wang, S., Wang, X., Bent, G. "Audio analysis as a control knob for social sensing," in Proc. of the 2nd International Workshop on Social Sensing (SocialSens'17), Apr. 2017.

[6] Wood, D., Wang, S., Salonidis, T., Conway-Jones, D., Ko, B.J., White, G., "Distributed analytics for audio sensing applications," Proc. SPIE 10635, Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR IX, 106350P (4 May 2018).

[7] Ko, B.J., Ortiz, J., Salonidis, T., Touma, M., Verma, D., Wang, S., Wang, X., Wood, D. "Demo abstract: acoustic signal processing for anomaly detection in machine room environments,"in Proc. of ACM BuildSys 2016.

[8] Agrawal, D. et. al. Challenges and opportunities with big data. https://docs.lib.purdue.edu/cctech/1

[9] Cohn, D., Ghahramani, Z., Jordan, M.I, . "Active learning with statistical models," in *Journal of artificial intelligence research* 4 (1996): 129-145.

[10] I. Goodfellow, et. al. "Generative adversarial nets." In *Advances in neural information processing systems*, pp. 2672-2680. 2014.

[11] Doersch, C., "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908* (2016)