Demo Abstract: Interactive Building Metadata Normalization

Jason Koh*, Kuo Liang*, Yiming Yang*, Dezhi Hong*, Yuvraj Agarwal†, Rajesh Gupta*,
{jbkoh,kuliang,yiy001,dehong}@ucsd.edu,yuvraj@cs.cmu.edu,rgupta@ucsd.edu
*University of California, San Diego, †Carnegie Mellon University

ABSTRACT

Having standardized metadata is the first step toward deploying smart building applications over heterogeneous buildings. Such a conversion process is highly manual because of different conventions in existing building metadata and diverse building configurations. Many machine learning methods have been attempted to ease the process by reducing the amount of experts' training examples and reusing the knowledge in different data sets. However, many of the end-users, such as building managers and commissioning practitioners, are unfamiliar with machine learning and programming interfaces. We implement and demonstrate a web-based graphical user interface whose workflow is designed based on a common programming interface, Plaster, for building metadata normalization. We implement three algorithms, Zodiac, BuildingAdapter, and Scrabble, though any new algorithms can be added. Users are instructed for proper actions with information visualization at each step to easily complete the procedure. The service is freely available at https://plaster.ucsd.edu.

KEYWORDS

smart buildings, metadata, machine learning, HCI

ACM Reference Format:
Jason Koh*, Kuo Liang*, Yiming Yang*, Dezhi Hong*, Yuvraj Agarwal†, Rajesh Gupta*, . 2019. Demo Abstract: Interactive Building Metadata Normalization. In *The 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys '19), November 13–14, 2019, New York, NY, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3360322.3360990

1 INTRODUCTION

Standard metadata schemata provide a common entity discovery mechanism to different apps, which is a key for large scale applications (apps) deployment. Several metadata schemata have been proposed such as Project Haystack and Brick metadata schema [1] to standardize the representation of entities in buildings and querying mechanisms. However, instantiating such metadata schema demands a lot of human effort as well as domain expertise. Buildings are heterogeneous; they are manually configured by different vendors with little standard. Metadata schemata have different ways of annotating context of entities with specific vocabulary sets. Thus, a domain expert needs to manually convert existing building metadata according to a standard schema often by a set of rules, which does not scale across different buildings and even in a building.

Machine learning algorithms can ease the process by reducing the number of required training examples or reusing the knowledge learned from buildings' metadata for other buildings. For example, Scrabble [3] can effectively learn the common features across different buildings to reduce the required training samples. BuildingAdapter [2] reuses the timseries features learned in a building to interpret another building's metadata. While these algorithms are specialized for their own optimization goals, they share common components such as a workflow, data models, and programming interfaces. Plaster [4] standardizes the common components and individual algorithms just need to implement the common programming interface. It is analogous to `scikit-learn` that defines a generic interface over different machine models in Python. This enables benchmarks and integration of different algorithms, and agile development of new ones.

While Plaster standardizes programming components to ease programmers' effort on dealing with different algorithms, users including building managers and commissioning practitioners might be unfamiliar with programming environments. We, thus, implement and demonstrate a Web-based Graphical User Interface (GUI), Plaster UI, guiding users to easily follow the workflow of metadata normalization with different algorithms fitting their requirements. It is automatically adapted to the user's configuration highlighting only the necessary information for each configuration. The service is freely available as well as the code base is open-sourced[1].

2 METADATA NORMALIZATION

Metadata normalization is a task to extract standard metadata from unstructured information. Fig. 1 shows an example of how an entity's information is normalized into a standard format such as Brick [1]. There are various types of available information such as timeseries data and raw metadata including entity names and units (Fig. 1(a)). Each of the information types represents a certain context of an entity such as measurement types and sensor locations. Diverse machine learning algorithms use such information to identify various types of structured metadata as in Fig. 1(c). `RM-101.T` is an instance of `temperature sensor` and it implies the associated location, `RM-101`. Some algorithms are optimized for identifying entity types only [2] while others can identify all the entities and their relationships from the information about an entity [3]. Different algorithms also need different types of labels as in Fig. 1(b).

For all the components in the process, Plaster [4] defines a common programming interface in Python as well as a unified database

[1]Web Service: https://plaster.ucsd.edu Code: https://github.com/plastering/plaster-ws

BuildSys '19, November 13–14, 2019, New York, NY, USA

ACM ISBN 978-1-4503-7005-9/19/11.
https://doi.org/10.1145/3360322.3360990

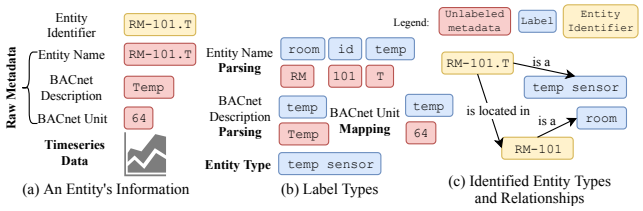

Figure 1: An Example Normalizing an Entity's Metadata.

model for all the necessary data types. This enables benchmark and integration of different algorithms and eases the process to develop a new algorithm. However, due to the diversity of the algorithms, end-users, such as building managers and commissioners, need better guidance in actual user interfaces.

3 PLASTER USER INTERFACE

Plaster User Interface (UI) implements the functions required in the original Plaster package in a workflow. The canonical functions in Plaster are `insert_examples`, `select_examples`, `update_model`, and `infer`. Their details are found in [4]. These functions are used throughout our workflow, whose steps are following:

(1) *Task Configuration*: A user chooses an algorithm or a combination of algorithms based on their optimization targets. The user also needs to load data.
(2) *Interactive Labeling*: A user provides examples for metadata normalization. Algorithms can choose the most informative examples with non-redundant patterns. It improves the sample efficiency reducing the human effort eventually. This is an iterative process that involves training models, requesting new examples, and submitting labels.
(3) *Result Review*: Once enough examples are given in the previous stage, the user can visually review the result of metadata normalization in a graph or a table. The user can also export the result to use it in external systems.

In the workflow, Interactive Labeling is the most complicated process as users need to iteratively interpret visualized information and provide corresponding labels. Fig. 2 shows the visualization, and the caption describes each component in detail. While we provide the flexibility to revoke the actions, we visually guide users to only the required actions among many possibilities. For example, in Fig. 2, the entity's type has been provided, so the insertion action is deactivated by showing "Inserted", and "Next" is activated.

4 DISCUSSIONS

We have implemented and demonstrated how the user should interact with machine learning algorithms to most efficiently normalize building metadata. With Plaster UI, even though the users are not familiar with machine learning models and standard schemata, they can convert their buildings into the standard format such as Brick. Better accessibility to standard schemata will eventually expedite the adoption of smart building applications.

While Plaster UI optimizes user interaction patterns, machine learning models still need to be more scalable in terms of heterogeneity of both data and labels. Different buildings have data with different styles and users would want to normalize them altogether.

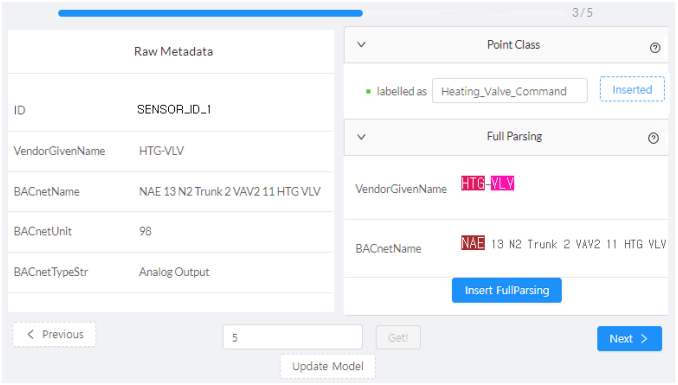

Figure 2: A Screenshot of Plaster UI. The bar and the ratio at the top of the screen show the progress of the current round. The left panel visualizes different types of Raw Metadata. At the right panel, users can interactively provide different types of labels. Users may select a customized number of examples to label in a round with Get button, and move to different examples by Next/Previous buttons within the round. Update Model trains the machine learning model based on the training examples labeled so far. Blue buttons are guided actions for the user, indicating recommended next steps. White buttons indicate the actions have been finished but the user may revoke them. Gray buttons are for inactivated actions at the current status.

Furthermore, multiple users may want to collaborate each other for normalizing a set of buildings too. This crowd-sourced labeling process often incurs inconsistency in labels because of different understandings of the data and human errors. Thus, underlying machine learning algorithms should be tolerant of the disagreeing examples, which could be mitigated by introducing a weakly supervised machine learning framework across different users and models [5].

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1526841 and 1526237.

REFERENCES

[1] Bharathan Balaji, Arka Bhattacharya, Gabriel Fierro, Jingkun Gao, Joshua Gluck, Dezhi Hong, Aslak Johansen, Jason Koh, Joern Ploennigs, Yuvraj Agarwal, et al. 2018. Brick: Metadata schema for portable smart building applications. *Applied energy* 226 (2018), 1273–1292.
[2] Dezhi Hong, Hongning Wang, Jorge Ortiz, and Kamin Whitehouse. 2015. The building adapter: Towards quickly applying building analytics at scale. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. ACM, 123–132.
[3] Jason Koh, Bharathan Balaji, Dhiman Sengupta, Julian McAuley, Rajesh Gupta, and Yuvraj Agarwal. 2018. Scrabble: transferrable semi-automated semantic metadata normalization using intermediate representation. In *Proceedings of the 5th Conference on Systems for Built Environments*. ACM, 11–20.
[4] Jason Koh, Dezhi Hong, Rajesh Gupta, Kamin Whitehouse, Hongning Wang, and Yuvraj Agarwal. 2018. Plaster: An integration, benchmark, and development framework for metadata normalization methods. In *Proceedings of the 5th Conference on Systems for Built Environments*. ACM, 1–10.
[5] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment* 11, 3 (2017), 269–282.