DreamCreek: AI for Battery Formation and Grading

Wenfei Fan[1,2,3], Yang Leng[1], Daji Li[1], Shuhao Liu[1], Mingliang Ouyang[1], Yaoshu Wang[1], Min Xie[1], Qiang Yuan[1]
[1]*Shenzhen Institute of Computing Sciences* [2]*University of Edinburgh* [3]*Beihang University*
wenfei@inf.ed.ac.uk, {lengyang, lidaji, shuhao, ouyangmingliang, yaoshuw, xiemin, yuanqiang}@sics.ac.cn

***Abstract*—DreamCreek is a system for grading the capacity of lithium-ion battery cells. An electric vehicle (EV) battery pack consists of thousands of lithium-ion cells; these cells must have a balanced capacity, measured by a formation and grading phase. This phase is costly, taking 14–20+ hours. DreamCreek aims to optimize this process, by collecting data from partial charge. Using the data, it determines the capacity of lithium-ion cells by employing both machine learning prediction and logic deduction, to reduce the usage of energy and increase the production. We will demonstrate how DreamCreek works with a guided tour, and show how it reduces the time of the formation and grading phase to 4 hours, with an error rate in the range of [0.06%, 1%].**

***Index Terms*—Battery formation and grading; graph analysis**

I. Introduction

The Battery Formation and Grading System Market has been singled out as a specialized industry segment [1] for the manufacturing and maintenance of batteries, including those used in electric vehicles (EVs), electronics, and energy storage.

For example, an EV battery pack is composed of thousands of lithium-ion cells. These cells must have a balanced capacity, to avoid safety issues like thermal runaway. A key process to the manufacturing of the cells has two steps: (a) it starts with a formation step to "activate" a cell by performing initial charge on it, followed by a battery standing step to stabilize its voltage and cool off the cell, and (b) it then conducts a grading step to measure the capacity of the cell based on its individual characteristics, by continually charging until fully charged and then fully discharging it. The total amount of electricity discharged is the capacity of the cell. This process is essential to ensure the reliability and quality of the cell.

This process is, however, costly. It takes 14 to 20+ hours at different production lines in the industry. It is energy-consuming; as estimated in [2], reducing the battery charge energy by half can save $65,000 per GWh. It accounts for more than one third of the total cost of manufacturing, and is considered one of the bottlenecks that hinder manufacturers from increasing output and reducing the cost of battery production.

In light of this, the industry calls for AI technologies to help [1]. ML models have been tried to predict the battery capacity. However, the accuracy of the ML models often struggles to meet the ultra-high requirements ($\leq 1\%$ error rate) in battery manufacturing. Is it possible to improve by reducing the false positive (FPs) and false negatives (FNs) of ML predictions?

Dream Creek (Section II). In response to the needs, we have developed a system, DreamCreek, to assist battery formation and grading. It has been deployed at several production lines; it reduces the time of formation and grading to 4 hours, and keeps the error rate under 1%, as low as 0.06%. These translate to an 80% reduction in energy consumption.

DreamCreek has the following unique features.

Partial charge. As opposed to full charge and discharge in traditional process, DreamCreek only partially charges a cell, *e.g.,* to 40% (or to a cutoff voltage), and collects data in that process. It then grades the cell using the data without any discharge. By analyzing the data (including those collected before formation), DreamCreek accurately grades the capacity.

ML + logic. The core of DreamCreek is the graph association rules (GARs) [3], which consists of a graph pattern and a logic dependency. The pattern captures the association between procedures, states and metadata of a cell, and the dependency deduces the capacity based on the association from the data collected. GARs may embed ML models for grading and anomaly detection, and are discovered from historical data offline.

Grading and anomaly detection. DreamCreek collects data online during manufacturing, based on which it determines the capacity of a cell or its anomaly (*i.e.,* it cannot be packed with other cells). By training and embedding designated ML models in GARs, it filters FPs and FNs of ML predictions by incorporating additional logic conditions in GARs [4], and reduce the need for large amount of training data for ML.

Flexibility. We can configure parameters in DreamCreek, *e.g.,* the combination of procedures and constraints of a procedure, and adapt it to different production lines. Here each production line has its cutoff voltage (which decides the ratio $\alpha\%$ of partial charge); this is determined by its manufacturing process.

Demonstration (Section III). We will provide a guided tour of DreamCreek. We will showcase how capacity is graded and how anomaly is detected, reducing the time and energy of battery manufacturing. Participants are invited to interact with DreamCreek and experience how it beats traditional processes.

II. An Overview of Dream Creek

This section presents the background of the battery formation and grading process (Section II-A), graph association rules (Section II-B) and the system architecture (Section II-C).

A. Battery Grading

There are various battery-manufacturing production lines, each consisting of different procedures (see Figure 1 for an example production line), *e.g.,* stir, formation, grading, executed one by one. No matter what production line is used, formation and grading make a key phase. The capacity of a cell is not a constant due to the fluctuation of production parameters.

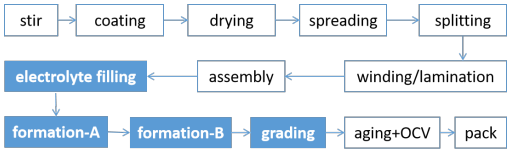

Fig. 1: A diagram of one battery cell production flow

To grade the capacity of a cell, the traditional formation and grading phase is costly (*e.g.,* 14 to 20+ hours) since it requires full charge/discharge (see Section I). Worse yet, this phase is hard to speed up physically, *e.g.,* the charging current must not exceed the *recommended charge rate* for safety reasons.

In light of this, ML models have been explored to replace the actual tests of capacity [5], [6]. As reported by our industry partners, however, these models often yield a high error rate, since it is challenging to handle th high variability in battery chemistry (*e.g.,* a slight difference in temperature can greatly impact the capacity) and the lack of adequate and representative training data that accurately reflects these variable factors.

B. AI for Grading

To improve the accuracy, DreamCreek takes all procedures of the production process into account, collects data, partially charges the cells in formation and grading, and deduces the capacity by unifying ML prediction and logic deduction.

Graphs. DreamCreek discretizes the production of cells as a time series of procedures, namely $p_{t_1}, \ldots, p_{t_m}$, and models the process as a graph $G = (V, E, L)$, so that we can capture the interactions and associations between different stages of the manufacturing process, allowing for better analysis, especially for skewed training data that lacks adequate samples at tail.

More specifically, (1) V is a finite set of vertices and each vertex v in V can be a procedure, a state of procedures or a battery cell; (2) $E \subseteq V \times L(E) \times V$ is a finite set of edges (v, l, v') with label l; we support three types of edges: (a) a transition edge connects two procedures (p_{t_i}, p_{t_j}) for $t_i < t_j$, (b) an association edge links a battery to a procedure, and (c) a status edge connects a procedure p and a state; and (3) each vertex v in V can also carry attributes along with range constraints, *e.g.,* a procedure vertex can carry its charging current range, and a battery vertex can carry its capacity interval. An example piece of G is shown in Figure 2.

GARs. DreamCreek grades the capacity by applying Graph Association Rules (GARs) [3]. GARs are defined with (a) a graph pattern to collect relevant entities (*e.g.,* procedures and states), and (b) a dependency on the correlations of the entities.

Graph pattern. A *pattern* is $Q[\bar{x}] = (V_Q, E_Q, L_Q, \mu)$, where (1) V_Q (resp. E_Q) is a set of vertices (resp. edges), (2) L_Q assigns a label $L_Q(u)$ (resp. $L_Q(e)$) to each $u \in V_Q$ (resp. $e \in E_Q$), (3) $\bar{x}$ is a list of distinct variables, and μ is a bijective mapping from $\bar{x}$ to V_Q, a distinct variable to each v in V_Q.

A *match* of $Q[\bar{x}]$ in graph G is a homomorphism h from Q to G such that (a) for each $u \in V_Q$, $L_Q(u) = L(h(u))$; (b) for each $e = (u, l, u')$ in Q, $e' = (h(u), l, h(u'))$ is an edge in G.

Dependency. A *predicate* of $Q[\bar{x}]$ is one of the following:

$$p ::= x_i.A \otimes x_j.B \mid x_i.A \otimes c \mid \mathcal{M}(x_1.\bar{A}_1, \ldots x_n.\bar{A}_n),$$

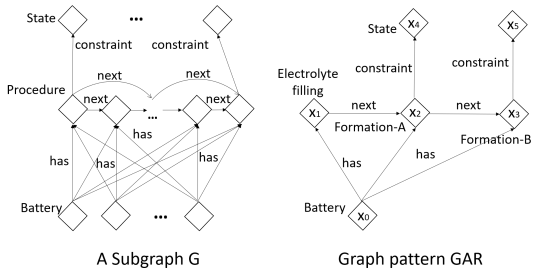

Fig. 2: Example graph and pattern

where $\otimes$ is one of $=, \neq, <, \leq, >, \geq$; x_i is a variable in $\bar{x}$ ($i \in [1, n]$); c is a constant; A and B are attributes; and $x_i.\bar{A}_i$ is a list of attributes at "vertex" x_i. We support (a) *attribute predicates* (*i.e.,* $x_i.A \otimes x_j.B$ and $x_i.A \otimes c$) to compare values; and (b) *ML predicates* that return a Boolean value at $(x_1.\bar{A}_1, \ldots, x_n.\bar{A}_n)$, *e.g.,* $\mathcal{M} < \delta$ if the predicted capacity of a cell is less than a given threshold δ for a regression model $\mathcal{M}$.

A *Graph Association Rule* (GAR) φ is defined as

$$Q[\bar{x}](X \to p_0),$$

where p_0 is either $x_0.\mathsf{capacity} = c$ for capacity grading of battery x_0, or $x_0.\mathsf{status} = \mathsf{anomalous}$ for anomaly detection, $Q[\bar{x}]$ is a pattern, X is a conjunction of predicates. We refer to $Q[\bar{x}]$ and $X \to p_0$ as the *pattern* and *dependency* of φ, and to X and p_0 as the *precondition* and *consequence*, respectively.

Embedding ML in logic. DreamCreek trains several ML models using historical data, including (a) $\mathcal{M}_\mathsf{g}$, a LightGBM-based regression model [7] via supervised learning for capacity grading; and (b) $\mathcal{M}_\mathsf{a}$, a LightGBM-based classification model, which returns true if it predicts a cell to be anomalous.

As remarked earlier, using ML models alone often yields a high error rate, *e.g.,* given inaccurate $\mathcal{M}_\mathsf{a}$, it may easily misclassify a cell. To tackle this, ML models are embedded as predicates in GARs, and we use additional conditions to reduce FPs and FNs of ML models, elaborated using $\mathcal{M}_\mathsf{a}$ as follows.

(1) $Q[\bar{x}](\mathcal{M}_\mathsf{a} = \mathsf{true} \wedge X_1 \to x_0.\mathsf{status} \neq \mathsf{anomalous})$. That is, although $\mathcal{M}_\mathsf{a}$ predicts the cell x_0 to be anomalous (*i.e.,* predicts true), if X_1 holds, we override the prediction of $\mathcal{M}_\mathsf{a}$ and conclude that x_0 is not anomalous, reducing the FPs of $\mathcal{M}_\mathsf{a}$.

(2) $Q[\bar{x}](\mathcal{M}_\mathsf{a} = \mathsf{false} \wedge X_2 \to x_0.\mathsf{status} = \mathsf{anomalous})$. That is, while $\mathcal{M}_\mathsf{a}$ suggests that the cell x_0 is regular (*i.e.,* predicts false), we check additional conditions X_2 to filter FNs, *i.e.,* it is still detected as anomalous if conditions in X_2 are satisfied.

Note that we also use $Q[\bar{x}](X_3 \to x_0.\mathsf{status} = \mathsf{anomalous})$, where X_3 includes no ML predicate, to detect anomaly based on side features that are overlooked by ML model $\mathcal{M}_\mathsf{a}$.

Examples. Below are (simplified) GARs from real-life data.

(1) Capacity grading. A (simplified) GAR is $\varphi_1 = Q_1[\bar{x}](X_1 \to x_0.\mathsf{capacity} = 8)$, where Q_1 is a pattern shown in Figure 2, and its consequence grades a matching battery cell as Capacity Interval 8. Together with pattern Q_1, precondition X_1 specifies the following: (a) the weight before and after the Electrolyte Filling procedure (x_1) is 555 ± 25g and 605 ± 25g, respectively; (b) its Formation-A procedure (x_2) uses a constant charging

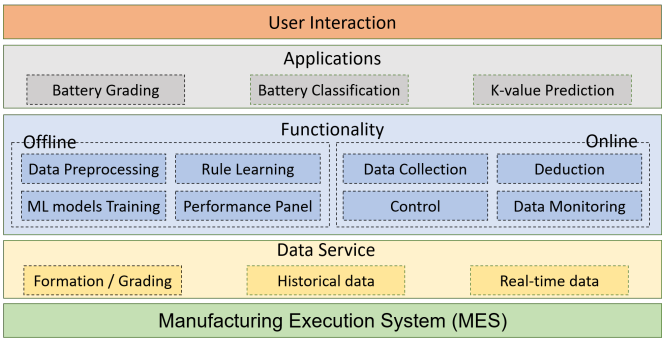

Fig. 3: The architecture of Dream Creek

current at 3.8A with initial voltage between 0–100mV, constrained by final state 324 (x_4); and (c) its Formation-B procedure (x_3) uses a constant charging current at 8.8A, with initial voltage between 3.3–3.4V, constrained by final state 738 (x_5).

(2) Anomaly detection. A GAR is $\varphi_2 = Q_2[\bar{x}](x_1.\mathsf{voltage} \leq 1.9\mathrm{V} \wedge x_1.\mathsf{temperature} \geq 86^{\circ}\mathrm{C} \wedge \mathcal{M}_{\mathsf{a}}(x_1) = \mathsf{false} \rightarrow x_0.\mathsf{status} = \mathsf{anomalous})$, where Q_3 is similar to Q_1 but does not contain state vertices. It overrides incorrect rulings of $\mathcal{M}_{\mathsf{a}}$ by considering erratic measurements (*e.g.,* voltage and temperature) in Electrolyte Filling (x_1), reducing the FNs of $\mathcal{M}_{\mathsf{a}}$.

(3) Regular cell identification. We can identify regular cells with side features, *e.g.,* a GAR is $\varphi_3 = Q_3[\bar{x}](x_2.\mathsf{voltage} \leq 3.30\mathrm{V} \wedge x_2.\mathsf{step\_time} \geq 269\mathrm{s} \wedge x_3.\mathsf{voltage} \geq 3.37\mathrm{V} \rightarrow x_0.\mathsf{status} \neq \mathsf{anomalous})$, where Q_3 is the same as Q_2. It identifies regular cells based the voltage and time spent on some steps in formation. This GAR was discovered from a production line in a leading manufacturer of electric vehicles.

Algorithms. DreamCreek implements the algorithms for GAR discovery [8] and association deduction with GARs [3]. Both algorithms are parallely scalable, *i.e.,* they provably guarantee to reduce runtime when given more processors [9].

Rule discovery. DreamCreek learns GARs and trains ML models (*e.g.,* $\mathcal{M}_{\mathsf{g}}$, $\mathcal{M}_{\mathsf{a}}$) offline using historical data. It discovers GARs for capacity grading and anomaly detection, by expanding pattern Q with procedure vertices level-wise and adding connected state vertices in the process. Meanwhile it learns dependencies on the attributes of these vertices [8]. The learned ML models are embedded as predicates in the mined GARs.

Association deduction. DreamCreek predicts capacity and detects anomaly online for each cell in production. Modeling the data as a graph G, it recursively applies the learned GARs to G by extending the chase [3]. It returns either the capacity of the cell, or a flag if the cell is an anomaly. The chase is Church-Rosser, *i.e.,* it converges at the same result no matter what discovered GARs are used and in what order they are applied [3]. That is, the process returns a deterministic result.

C. The Architecture of Dream Creek

DreamCreek is built on top of Manufacturing Execution System (MES) (see Figure 3), which has been deployed by the industry to track battery production. For each battery cell, DreamCreek reads online its time-series data monitored by sensors and collected by MES. It performs an initial charge to active the cell for formation, cools it off, and then recharges it to $\alpha\%$ of its full charge in the grading stage. Based on the data, it grades the capacity and detects anomaly using GARs mined.

Workflow. DreamCreek has offline and online parts.

Offline. In the offline phase, we begin by collecting historical data from MES, followed by data cleaning and enrichment for data pre-processing. Historical data is partitioned into training data, for training models and discovering GARs, and validation data for parameter tuning. It has the following main modules.

- *Pre-processing*. This module fixes errors and outliers [10], [11], gathers data from anomaly cells and builds graphs.
- *Rule learning*. This module trains ML models and discovers GARs using the processed historical data [8]. To facilitate training, it extracts attributes by computing (a) the difference between two (shift) time series; and (b) the statistical information in a fix-sized window, *e.g.,* max, min and mean.

For each production line, the models and GARs are learned only once and used for all cells. This said, DreamCreek monitors changes to the distribution of time series and accumulates new data. If the changes reach a certain threshold, it triggers (incremental) training and learning offline in the backend.

Online. Once the GARs and the models are ready, DreamCreek proceeds to the online phase, where it first collects real-time data from MES and then applies GARs to grade capacity and detect anomaly. It has the following main modules.

- *Data collection*. This module transforms the real-time data from MES into a graph after quick cleaning [10].
- *Deduction.* This module deduces the capacity of cells and detects anomaly by applying learned GARs to the graph.
- *Control.* This module (a) aggregates statistics about battery cells in production, *e.g.,* capacity distributions of cells; (b) estimates the reduction of time for each cell; and (c) visualizes the performance difference with various baselines.

As remarked earlier, DreamCreek has been deployed at different production lines. It accommodates various scenarios by allowing users to configure it to different production lines.

III. Demonstration Setup and Plan

This section proposes our plan for demonstrating.

Demonstration setup. We will use a single machine powered by 16GB RAM and 8 processors with Intel(R) Xeon(R) Gold 5320 CPU @2.20GH. For the demonstration, we will upload the data in dump files exported from the MES system.

For training, we will use historical data from a traditional production line, with measurements throughout the formation–grading lifecycle. For testing, we will use data from a production line at which DreamCreek is deployed. It (partially) charges a cell and decides its capacity using the data collected.

A guided tour. We will walk the participants through the formation and grading of battery production with DreamCreek.

(1) Configuration. We will show how DreamCreek adapts to different production lines. The users may choose different

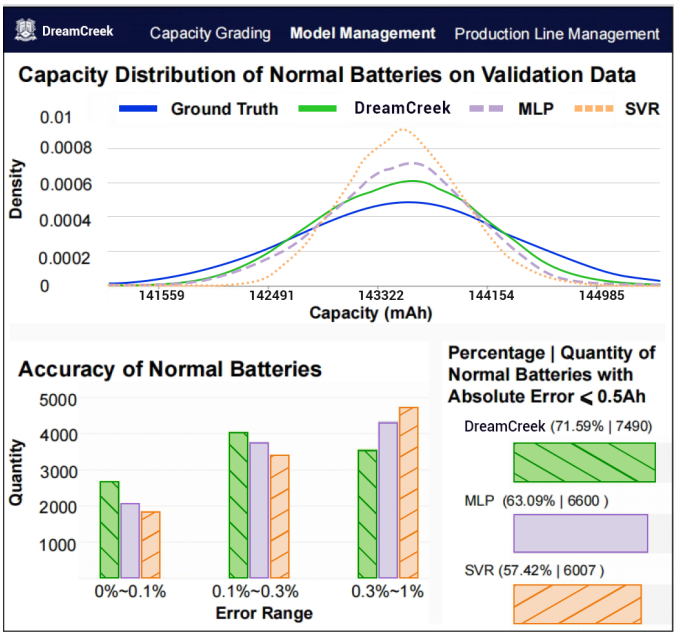

Fig. 4: A snapshot of analysis panel

combinations of procedures and set the parameters of each procedure. Moreover, the users may select ML models from a library, for capacity grading and anomaly detection.

(2) Offline processing. We will show how DreamCreek enriches historical data and uses the data to train the selected ML models and discovers GARs. We will showcase how the FPs and FNs of ML models are reduced in a validation dataset.

(3) Online processing. For each cell in production, we will show which part of the data is collected in partial charge. On the data, DreamCreek applies the discovered GARs, grades the capacity, and detects anomaly. We will explain the output of DreamCreek on each cell and estimate the reduction of time.

(4) Analysis panel. We will see how DreamCreek outperforms the existing baselines (see below). The analysis panel shows aggregated statistics about the graded cells, *e.g.,* their capacity distribution and the error rates compared with baselines.

Interaction. Participants can interact with DreamCreek.

(1) Capacity grading. The participants may pick a production line, and let DreamCreek determine the capacity of its cells.

In the analysis panel (Figure 4), they will see (a) the capacity distributions estimated by DreamCreek and other baselines (*e.g.,* the MLP and SVR models in [12]), (b) the real capacity distribution measured by full charge and full discharge, (c) the error rate comparison between different methods, and (d) the percentages of regular/anomalous cells (not shown). As shown there, the estimated capacity distribution of DreamCreek is close to the real distribution. In comparison, the capacity distributions of other methods deviate a lot from the real distribution, resulting in higher error rates and absolute errors.

(2) Anomaly detection. We will also invite the participants to experience the process of anomaly detection. They can view the analysis details of any selected cell, and examine the results

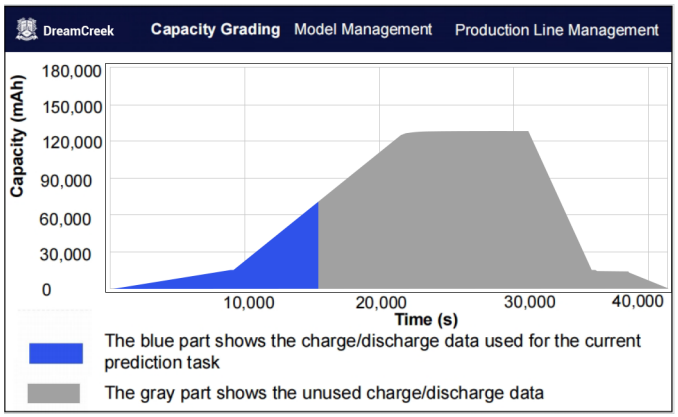

Fig. 5: A snapshot of capacity curve

returned by DreamCreek and by the traditional process.

The snapshot in Figure 5 visualizes the comparison between the two processes, where the x-axis represents the formation-grading lifecycle and the y-axis is the capacity of the selected cell. The traditional process requires full charge and full discharge. In contrast, DreamCreek significantly reduces the time bt collecting necessary (partial) data without discharge, based on which it detects anomaly in seconds. Users may compare the two processes, and experience how DreamCreek works.

(3) Others. The participants are also welcome to witness how DreamCreek collects data from MES, enriches the data with external features, trains the models, and discovers GARs.

Acknowledgment

The research of Dr. Min Xie was supported in part by Longhua Science and Technology Innovation Bureau 11501 A20240704A24CA6C and China NSFC 62202313.

References

[1] M. R. I. Lab, "Battery formation and grading system market size, outlook: Share, growth, and forecast (2024-2031)," 2024.
[2] Y. Yuan, X. Kong, J. Hua, Y. Pan, Y. Sun, X. Han, H. Yang, Y. Li, X. Liu, X. Zhou, L. Lu, and H. Wang, "Fast grading method based on data driven capacity prediction for high-efficient lithium-ion battery manufacturing," *Journal of Energy Storage*, vol. 73, 2023.
[3] W. Fan, R. Jin, M. Liu, P. Lu, C. Tian, and J. Zhou, "Capturing associations in graphs," *PVLDB*, vol. 13, no. 11, pp. 1863–1876, 2020.
[4] L. Fan, W. Fan, P. Lu, C. Tian, and Q. Yin, "Enriching recommendation models with logic conditions," *SIGMOD*, 2024.
[5] K. Liu, Z. Wei, Z. Yang, and K. Li, "Mass load prediction for lithium-ion battery electrode clean production: A machine learning approach," *Journal of Cleaner Production*, vol. 289, p. 125159, 2021.
[6] R. P. Cunha, T. Lombardo, E. N. Primo, and A. A. Franco, "Artificial intelligence investigation of NMC cathode manufacturing parameters interdependencies," *Batteries & Supercaps*, vol. 3, no. 1, 2020.
[7] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *NeurIPS*, 2017.
[8] W. Fan, W. Fu, R. Jin, P. Lu, and C. Tian, "Discovering association rules from big graphs," *PVLDB*, vol. 15, no. 7, pp. 1479–1492, 2022.
[9] C. P. Kruskal, L. Rudolph, and M. Snir, "A complexity theory of efficient parallel algorithms," *Theor. Comput. Sci.*, 1990.
[10] W. Fan, P. Lu, C. Tian, and J. Zhou, "Deducing certain fixes to graphs," *PVLDB*, vol. 12, no. 7, pp. 752–765, 2019.
[11] W. Fan, M. Liu, S. Liu, and C. Tian, "Capturing more associations by referencing knowledge graphs," *PVLDB*, 2024.
[12] D. G. Murray, J. Simsa, A. Klimovic, and I. Indyk, "tf.data: A machine learning data processing framework," *PVLDB*, 2021.