An In-Depth Correlative Study Between DRAM Errors and Server Failures in Production Data Centers

Zhinan Cheng[1], Shujie Han[2], Patrick P. C. Lee[1], Xin Li[3], Jiongzhou Liu[3], Zhan Li[3]
[1]The Chinese University of Hong Kong [2]Peking University [3]Alibaba Group

***Abstract*—Dynamic Random Access Memory (DRAM) errors are prevalent and lead to server failures in production data centers. However, little is known about the correlation between DRAM errors and server failures in state-of-the-art field studies on DRAM error measurement. To fill this void, we present an in-depth data-driven correlative analysis between DRAM errors and server failures, with the primary goal of predicting server failures based on DRAM error characterization and hence enabling proactive reliability maintenance for production data centers. Our analysis is based on an eight-month dataset collected from over three million memory modules in the production data centers at Alibaba. We find that the correctable DRAM errors of most server failures only manifest within a short time before the failures happen, implying that server failure prediction should be conducted regularly at short time intervals for accurate prediction. We also study various impacting factors (including component failures in the memory subsystem, DRAM configurations, types of correctable DRAM errors) on server failures. Furthermore, we design a machine-learning-based server failure prediction workflow and demonstrate the feasibility of server failure prediction based on DRAM error characterization. To this end, we report 14 findings from our measurement and prediction studies.**

I. Introduction

Servers in modern data centers typically use dynamic random access memory (DRAM) as main memory for fast data management. However, DRAM errors (i.e., bit errors in DRAM reads) are reportedly prevalent in production environments [23], [31], [32], [39], [43]–[47], due to magnetic and electrical interferences [4], [33], [36], operation disturbances [8], [25], and hardware wear-outs [5]. Although error correcting codes (ECC) are used in DRAM hardware to detect and correct DRAM errors, DRAM errors still lead to *server failures* (i.e., the servers no longer support data management for host applications) if either (i) there exist too many erroneous bits in DRAM that exceed the correctable limit of ECC or (ii) there exist too many DRAM errors within a server that make the server fail to operate even though each such DRAM error is correctable by ECC. In production environments, the proportion of server failures caused by DRAM errors is substantial. For example, an earlier study shows that around 30% of hardware-related server failures in high-performance computing sites are caused by DRAM errors [42]; a more recent study shows that DRAM errors account for 3.06% of all hardware-related server failures in data centers, while such a percentage implies thousands of server failures and is statistically significant [48].

Field studies in the literature [23], [31], [32], [39], [43]–[47] have extensively analyzed the trends and statistical properties of DRAM errors in production environments, yet little is known about how DRAM errors are correlated with server failures. Such a limitation is mainly attributed to the unavailability of trouble tickets on server failures, thereby preventing existing studies from further characterizing the relationships between DRAM errors and server failures. We argue that understanding the correlation between DRAM errors and server failures is critical to proactive reliability maintenance; in particular, by characterizing DRAM errors before server failures happen, system administrators can proactively predict imminent server failures that are caused by DRAM errors and apply repair measures in advance (e.g., by relocating the data in the soon-to-fail servers to other healthy servers).

To elaborate, the following questions remain unexplored but are important for understanding the correlation between DRAM errors and server failures. (i) How do DRAM errors cause server failures? (ii) When and how common do DRAM errors occur before server failures happen? (iii) How do component failures with the manifestation of DRAM errors in the memory subsystem lead to server failures? (iv) How do DRAM configurations lead to server failures? (v) How do different types of DRAM errors manifest in DRAM before server failures happen? (vi) How can we accurately predict imminent server failures based on DRAM error characterization?

In this practical experience report, we present an in-depth data-driven analysis to study the correlation between DRAM errors and server failures. We conduct our analysis from the server failure prediction perspective, with the primary goal of achieving accurate server failure prediction based on DRAM error characterization. Specifically, we conduct a measurement study on the production data centers at Alibaba. We study a dataset collected from 250 K servers with more than 3 M Dual In-line Memory Modules (DIMMs) over an eight-month period. Our dataset covers DRAM error logs, trouble tickets of server failures caused by DRAM errors, and inventory logs. We make the following key contributions:

- We study the characteristics of correctable DRAM errors in 2,137 server failures that are caused by DRAM errors. We find that in most server failures, correctable DRAM errors only manifest within a short time before the failures happen (e.g., within one hour for more than 40% of server failures). If we conduct server failure prediction less frequently, many such server failures become unpredictable as they do not show any DRAM error symptoms.
- We analyze the impact of different aspects of DRAM errors on server failures, including component failures in the memory subsystem, DRAM configurations, and types of correctable DRAM errors.

- We design a machine-learning-based server failure prediction workflow that incorporates the characteristics of correctable DRAM errors into feature generation. Based on our measurement findings, we study how the choices of feature generation, prediction models, and prediction parameters affect the prediction accuracy. Our workflow achieves up to 62.9% of F1-score in the prediction of server failures that are driven by correctable DRAM errors, and also reduces thousands of hours of server downtime.

To this end, we report 14 findings from our measurement and prediction studies. We release our dataset at https://github.com/alibaba-edu/dcbrain/tree/master/dramdata and all the source code at https://github.com/zncheng/dramanalysis.

II. Background of DRAM

DRAM organization. DRAM is formed by a group of DIMMs. Specifically, each server in our data center has two processor *sockets*, each of which is connected to two integrated *memory controllers*. Each memory controller is further connected to three *channels* with up to two DIMMs each. Thus, each server can be equipped with a total of up to 24 DIMMs.

At a low level, a DIMM comprises multiple DRAM *chips* grouped in *ranks*, such that the chips in the same rank can be simultaneously accessed in a DRAM read/write. Each chip contains multiple *banks* that can be operated in parallel. Each bank is further partitioned into *rows* (e.g., 64 K) and *columns* (e.g., 1 K), and the element in a row and a column is called a *cell*, which can store a single bit of data.

A chip can have four or eight *pins*, referred to as x4 or x8 chips, respectively. Each rank forms 64 pins, either in 16 x4 or in eight x8 DRAM chips, that are connected to a 64-bit data bus for accessing data in 64-bit units (called *words*).

DRAM errors. We define a *DRAM error* as a symptom when DRAM behaves abnormally, such that one or multiple bits in DRAM are read differently from what they were written. Modern DRAM adopts error correcting codes (ECC) to protect against DRAM errors. Representative examples of ECC include single-error-correction-double-error-detection (SEC-DED) [10] for correcting any one flipped bit and detecting any two flipped bits in a single word, as well as single device data correction (SDDC) [1], [9] (or Chipkill ECC [11]) for correcting up to four bits in a single word. Most servers in our data centers use SDDC, except that servers of DRAM model B3 use SEC-DED (Section III). To detect and correct any DRAM errors by ECC, the memory controller scans the DRAM cells periodically based on memory scrubbing [3]. Scrubbing in our data centers is carried out every 24 hours.

If the number of erroneous bits in a DRAM error exceeds the correctable limit of ECC, the DRAM error cannot be fixed. We classify a DRAM error as: (i) a *correctable error (CE)* if it can be corrected by ECC, or (ii) an *uncorrectable error (UE)* if it can be detected but cannot be corrected by ECC.

Server failures. In this work, we focus on the *server failures* that are caused by DRAM errors, such that the failed servers can no longer support the normal DRAM access operations for their hosted applications. Such failures are reported in trouble tickets (Section III) by our maintenance system based on a set of pre-configured rules. Specifically, we classify server failures into three *failure types*:

- *UE-driven failure*: It refers to a server failure due to the occurrence of UEs, in which the server crashes or cannot allow its hosted applications to access data in DRAM. Once a server shows a UE, our maintenance system immediately treats the server as failed and reports a UE-driven failure.
- *CE-driven failure*: It refers to a server failure due to the occurrence of an overwhelmingly large number of CEs that cannot be properly handled by the server. Our maintenance system reports a CE-driven failure if the server has at least 10 K CEs in the past 24 hours. A CE-driven failure can be viewed as a denial of service attack [39] against the server.
- *Miscellaneous failure*: It refers to a server failure that is intricately related to DRAM errors, such as when too many DRAM pages are offline (e.g., at least 100 MB of offline pages) or some DIMMs are disconnected from the server and cannot be accessed by host applications. In such cases, our maintenance system reports a miscellaneous failure even though they are not directly driven by UEs or CEs.

III. Dataset

We collected data that records the DRAM errors and server failures from 250 K servers deployed in production data centers at Alibaba. Our dataset spans eight months; note that we anonymize the exact dates to prevent the memory models from being inferred based on the deployment dates. Our dataset includes three data types: DRAM error logs, trouble tickets on server failures, and inventory logs.

DRAM error logs. Each server collects its DRAM error logs using `mcelog` [27], a Linux tool that collects DRAM errors based on the Machine Check Architecture (MCA) [26]. It runs a `mcelog` daemon to record the details of any DRAM error event, including the server ID, DIMM ID, rank ID, bank ID, row ID, column ID, and detecting source (i.e., whether the error is detected by the memory scrubber or a DRAM read/write operation). Specifically, when a memory controller detects a DRAM error, it stores the error details in the hardware registers of MCA. Then the memory controller interrupts the CPU to notify both the operating system to handle the error and the `mcelog` daemon to record the error event.

In total, our DRAM error logs include 75.1 M CEs from 30,496 servers (including both healthy and failed servers) and 87,186 write errors from 351 servers over the eight-month span. In our analysis, we only focus on the CEs in the DRAM error logs, since the write errors are much fewer and they do not lead to server failures in our dataset. Note that UEs are not collected here, but instead in trouble tickets since the occurrence of a UE leads to a server failure.

Trouble tickets. Each server runs a background monitoring daemon that monitors system-level abnormal events (e.g., server crash) and sends system event logs to our centralized maintenance system, which checks for any server failure via rule-based detection. If a server failure is detected, our maintenance system issues a *trouble ticket*, which records the server ID,

Manufacturers	Servers%	8 DIMMs	12 DIMMs	16 DIMMs	24 DIMMs
M1	23.6%	4.1%	12.1%	0.9%	6.5%
M2	51.9%	6.4%	32.7%	1.4%	11.3%
M3	21.5%	3.6%	14.2%	0.6%	3.1%
M4	3.1%	0.3%	2.7%	0.0%	0.0%

TABLE I: Server population: percentages of servers from each manufacturer ("Servers%") and percentages of servers for each number of DIMMs attached to each server over the server population.

DRAM model	Capacity	Chip width	Servers%
A1	32 GB	x4	31.4%
A2	16 GB	x4	30.0%
B1	32 GB	x4	15.2%
B2	16 GB	x4	0.2%
B3	16 GB	x8	4.5%
C1	32 GB	x4	13.3%
C2	16 GB	x4	6.4%

TABLE II: DRAM populations: "Servers%" shows the percentages of servers for each DRAM model over the server population.

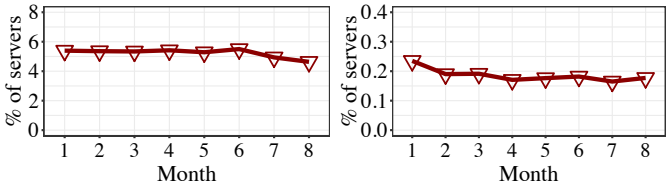
timestamp, and failure type. Our system administrators will further manually validate the trouble tickets and close them once they are fixed (e.g., the failed DIMM is replaced).

We finally collected a total of 3,017 trouble tickets over the eight-month span. Each trouble ticket corresponds to a server failure associated with a distinct server. Among all trouble tickets, 2,137 of them show at least one CE before the server failure, while the remaining ones do not show any CE before the server failure (e.g., some of them only show a UE). Since our study aims to analyze whether server failures are predictable via the characterization of CEs (Section IV-B), unless otherwise specified, we only focus on the 2,137 trouble tickets in which server failures are preceded by at least one CE. Among these 2,137 trouble tickets, there are 567 UE-driven failures, 809 CE-driven failures, and 761 miscellaneous failures.

Inventory logs. We further collected the product details about the DIMMs and servers in our dataset, so as to analyze the correlation between server failures and DRAM hardware configurations (Section V-B). Our servers are from four manufacturers (denoted by M1, M2, M3, and M4). Table I shows the percentages of servers from each manufacturer, as well as the percentages of servers for each number of DIMMs attached to a server. M2 accounts for the largest fraction (51.9%) of servers among all manufacturers, and most of the servers (61.7%) are equipped with 12 DIMMs.

Our dataset covers seven DRAM models from three major vendors. Due to proprietary concerns, we denote each DRAM model by "Vendor"-"k", where "Vendor" is represented by a letter ('A', 'B', and 'C') and "k" means the k-th model in the same vendor. Table II shows the percentages of servers and the hardware characteristics (e.g., capacities and chip widths). All DIMMs in a server are from the same DRAM model.

Limitations. We do not consider silent data corruptions [12] as well as the impact of application workloads and the absolute server age on DRAM errors, as such information is unavailable in our dataset. We also do not consider the security exploits that cause DRAM errors (e.g., Rowhammer attacks [25] that trigger bit flips), as all applications running in our data centers are under centralized administration.



(a) Servers with CEs (b) Server failures

Fig. 1: Finding 1. Percentages of servers with CEs and server failures.

IV. Characterization of Server Failures

We start with the high-level characterization of server failures caused by DRAM errors. We first compare the overall distributions of server failures with those in prior work (Section IV-A) and present our analysis methodology (Section IV-B). We further measure how CEs occur before server failures (Section IV-C).

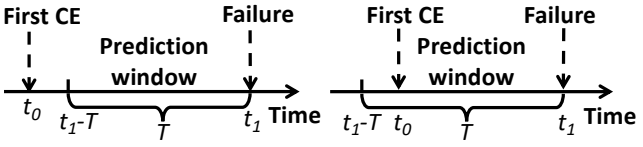
A. Overall Distributions

We first examine the overall distributions of servers with CEs and server failures over the eight-month span, and make comparisons with Facebook's study on DRAM errors published in 2015 [39]. Figure 1(a) shows the percentage of servers with CEs over all servers in each month. The percentage of servers with CEs stays stable with time, with 5.23% in each month on average. Overall, 11.8% of servers experience CEs; the percentage is slightly higher than 9.62% reported by Facebook [39]. One major reason of having a higher percentage of servers with CEs in our dataset is that our DIMMs have larger capacities (16 GB or 32 GB) than in [39] (e.g., 2-24 GB). Figure 1(b) shows the percentage of server failures over all servers in each month (we consider all 3,017 server failures here to calculate the overall failure rates). While our dataset includes 250 K servers, some servers may be added to or removed from production over time during the eight-month span. We observe that the percentage of server failures per month remains fairly stable across months, with an average of 0.19% (i.e., there exist hundreds of server failures per month). Overall, our results indicate that DRAM errors and the server failures caused by DRAM errors still pose reliability concerns to modern production data centers.

Finding 1. *Our dataset has a higher percentage of servers with CEs than Facebook's study [39], and the percentage of server failures per month remains fairly stable across months.*

B. Analysis Methodology

We conduct our measurement study from the server failure prediction perspective. Our goal is to characterize the manifestation of CEs before a server failure happens; by doing so, we can extract the proper features for model training and failure prediction based on machine learning (Section VI). Specifically, for each server failure, let t_0 be the time at which its first CE occurs, and t_1 be the time at which the server failure occurs. Let T be the *prediction window*, defined as the time interval prior to the server failure at which the failure prediction should be conducted. We say that a server failure is *predictable* if $t_0 \leq t_1 - T$ (Figure 2(a)), meaning that there



(a) Case 1: Predictable (b) Case 2: Unpredictable

Fig. 2: Predictable and unpredictable server failures based on the length of the prediction window.

exists some CEs for our feature extraction[1]; otherwise, a server failure is *unpredictable* if $t_0 > t_1 - T$ (Figure 2(b)), as no CE is observed when the prediction is conducted. Our analysis examines different types of CE manifestations by varying the length of T, so as to understand the predictability of a server failure. Note that the prior work [6] also defines the prediction window, but it focuses on the prediction of UEs, while we focus on the prediction of server failures.

In practice, we cannot exactly know when a server failure happens. We need to perform server failure prediction *periodically*, such that the periodic interval should be no larger than the prediction window in order to capture all possible predictable server failures with respect to the prediction window.

Our analysis treats the first CE observed in our dataset as the real first CE, but the real first CE may occur prior to the dataset collection. This leads us to falsely classify a predictable server failure that appears near the beginning of the dataset as unpredictable (i.e., no CE is observed in the dataset but some CEs exist prior to the dataset collection). Nevertheless, if we consider a small prediction window (as shown in our following analysis), the misclassification has limited impact. For example, if the prediction window is set to 30 minutes, we may falsely classify the server failures that appear in the first 30 minutes of the dataset, but they only account for 0.05% of all server failures in the whole dataset.

C. Characteristics of CEs Before Server Failures

We first examine the relative percentage of predictable server failures (i.e., there exists CEs before the prediction) for each failure type (i.e., CE-driven, UE-driven, and miscellaneous) by varying the prediction window from one minute to 30 days. Figure 3 shows that while large fractions of server failures are predictable in a short prediction window (e.g., 96.5%-99.9% when the prediction window is one minute), the relative percentages of predictable server failures decrease significantly as the prediction window increases. For example, the relative percentages of predictable CE-driven, UE-driven, and miscellaneous failures decrease to 31.8%, 33.5%, and 40.3% when the prediction window increases to one day, respectively. Thus, keeping a short prediction window is critical for accurate prediction. For example, if we choose a longer prediction window for predicting server failures (e.g., an hour), we will miss a significant fraction of server failures (e.g., 55.4% of CE-driven failures).

[1]Note that whether a server failure can be correctly predicted depending on the machine learning workflow (Section VI).

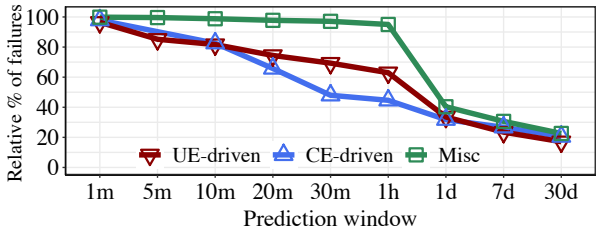


Fig. 3: Finding 2. Relative percentage of predictable server failures for each failure type versus the prediction window ('m' = minutes; 'd' = days).

Finding 2. *A significant fraction of server failures become unpredictable if the prediction window increases (e.g., beyond one day). Keeping a short prediction window (e.g., at the minute granularity) is necessary to achieve accurate server failure prediction.*

We next study the number of CEs that can be observed before the prediction is conducted. For each predictable server failure, we count the number of CEs before the prediction is conducted (i.e., the number of CEs outside the prediction window). We then compute the average number of CEs over all predictable server failures for each failure type. Figure 4 shows the average number of CEs before prediction for each failure type versus the prediction window. The average numbers of CEs for predictable CE-driven and miscellaneous failures are generally high; for example, they are 8.7-22.3 K and 5.5-10.1 K when the prediction window ranges from one minute to one day, respectively. Their numbers drop when the prediction window increases beyond seven days, due to fewer predictable server failures (Figure 3). On the other hand, the average number of CEs before prediction for predictable UE-driven failures is small; for example, it is only 333-518 when the prediction window ranges from one minute to one day.

We further study the CEs of healthy servers. The average number of CEs over all healthy servers in the whole eight-month period is 835, which is much less than those for predictable CE-driven and miscellaneous failures (note that we exclude three healthy servers with unexpectedly more than 3.9 M CEs). This suggests that the number of CEs can be an effective indicator for distinguishing between healthy servers and CE-driven or miscellaneous failures in server failure prediction. However, UE-driven failures have a limited number of CEs (comparable to that in healthy servers), and we need other features to aid our prediction.

Finding 3. *The number of CEs before prediction is significant in both predictable CE-driven and miscellaneous failures, implying that it can serve as an effective indicator for server failure prediction.*

We further study the frequency of CE occurrences before the prediction is conducted. For each predictable server failure, we measure the *mean time between errors (MTBE)*, defined as the average time between a pair of adjacent CEs before the prediction is conducted. We collect all MTBE samples across all predictable server failures for each failure type and plot the median one to avoid the extreme cases. Figure 5 shows the median MTBE for each failure type versus the prediction

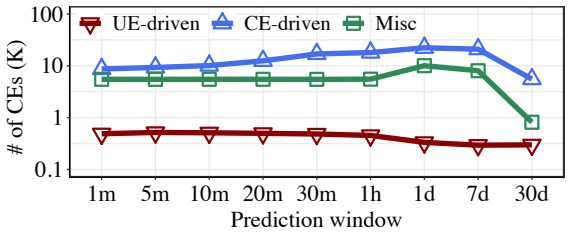


Fig. 4: Finding 3. Average number of CEs before prediction for each failure type versus the prediction window (y-axis is in log scale).

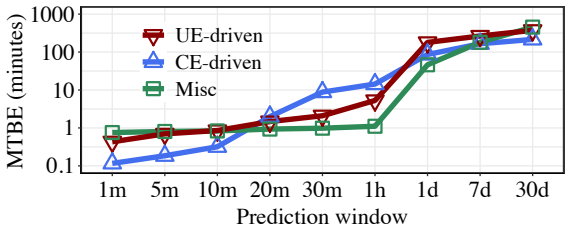


Fig. 5: Finding 4. Median mean time between errors (MTBE) before prediction for each failure type versus the prediction window (y-axis is in log scale).

window. The median MTBE values for all failure types are generally small in short prediction windows; for example, they are 5.3, 14.3, and 1.1 minutes for predictable UE-driven, CE-driven, and miscellaneous failures when the prediction window is one hour, respectively. Also, as the prediction window decreases, the median MTBE values become smaller. This suggests that the CEs tend to occur in bursts as a server failure is approaching.

Finding 4. *The MTBE is generally small in all predictable server failures in short prediction windows. Also, the MTBE decreases as the prediction window decreases.*

In summary, CEs typically occur within a short time before the prediction is conducted, and hence a large fraction of server failures become unpredictable when the prediction window is large. Thus, we need to use a short prediction window (e.g., at the minute granularity) to periodically predict server failures for accurate prediction.

V. Impacting Factors on Server Failures

To accurately predict server failures via machine learning, we need to generate effective features by understanding how different factors affect server failures. Thus, we analyze different impacting factors, including component failures in the memory subsystem, (Section V-A), DRAM configurations (Section V-B), and types of CEs (Section V-C), on server failures. Such impacting factors later are used as features in our machine-learning-based server failure prediction workflow (Section VI). In this section, we focus on the predictable server failures when the prediction window is five minutes, as system administrators in our case take at least five minutes to repair server failures (Section VI-A).

A. Component Failure Analysis

CEs can manifest in different components in the memory subsystem [39], [44]. Here, we examine the impact of different

Component	Condition
Socket failure	More than 1K errors across more than one channel in the same socket
Channel failure	More than 1K errors across more than one bank in the same channel, excluding above errors
Bank failure	More than 1K errors across more than one row, excluding above errors
Row failure	More than one column in the same row has errors, excluding above errors
Column failure	More than one cell in the same column has errors, excluding above errors
Cell failure	More than one error in the same cell within one minute, excluding above errors
Random errors	Remaining errors, excluding above errors

TABLE III: Categories of component failures in the memory subsystem [39].

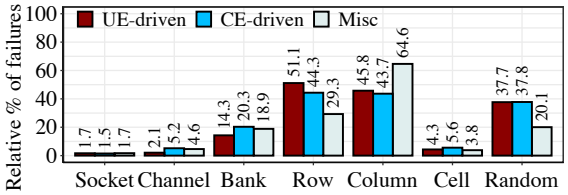


Fig. 6: Finding 5. Relative percentages of predictable server failures associated with component failures.

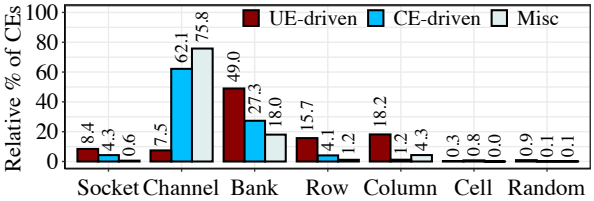


Fig. 7: Finding 5. Relative percentages of CEs due to component failures.

failed components in the memory subsystem on server failures. We consider six components based on the hierarchical architecture of the memory subsystem, namely sockets, channels, banks, rows, columns, and cells (Section II). Following the classification method in [39], we classify the CEs into different component failures, as shown in Table III.

We first examine the relative percentages of predictable server failures associated with different component failures. For each CE found in a server failure, we check which component failure that covers the CE. Given that a server failure may contain a large number of CEs, it may be associated with more than one component failure. Figure 6 shows that 94.1% of predictable server failures are associated with bank failures, row failures, column failures, or random errors in the memory subsystem. In particular, row failures account for 51.1% of UE-driven failures and 44.3% of CE-driven failures, while column failures account for 64.6% of miscellaneous failures.

We next examine the relative percentage of CEs across component failures. Figure 7 shows that channel and bank failures lead to high relative percentages of CEs. Specifically, channel failures have the highest relative percentages of CEs for CE-driven and miscellaneous failures (62.1% and 75.8%, respectively), while bank failures have the highest relative percentage of CEs for UE-driven failures (49.0%). However,

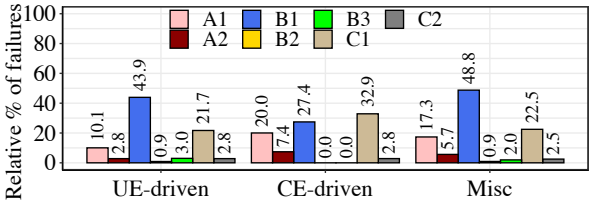


Fig. 8: Finding 6. Relative percentages of predictable server failures decomposed by the DRAM model (note that B2 and B3 do not have CE-driven failures).

DRAM model	A1	A2	B1	B2	B3	C1	C2
Population	**78 K**	72 K	**38 K**	0.5 K	11 K	**33 K**	16 K
Number of failure	**369**	127	**934**	13	35	**597**	62
Unpredictable failures	**18**	8	**92**	1	3	**37**	4

TABLE IV: Finding 6. Failure rates for different DRAM models.

we notice that the relative percentages of predictable server failures associated with channel and bank failures are less than 10% and 25%, respectively (Figure 6). The reason that both channel and bank failures have high percentages of CEs is that channels and banks are associated with a larger number of cells than rows and columns in the memory subsystem, meaning that any failed channel or bank can lead to a larger number of CEs than any failed row or column. For example, we find one extreme case with 3.7 M CEs due to one channel failure for CE-driven failures and seven extreme cases with at least 100 K CEs (up to 1.4 M CEs due to one channel failure) for miscellaneous failures. This suggests that a higher-level component in the hierarchical architecture of the memory subsystem has more severe impact on the number of CEs. Thus, DRAM chipmakers may need to improve the reliability of channels and banks.

Finding 5. *Most predictable server failures are due to bank failures, row failures, column failures, or random errors, while most CEs are associated with channel and bank failures.*

B. Impact of DRAM Configurations on Server Failures

We study how hardware configurations are correlated with server failures. We consider three scenarios: DRAM models, number of DIMMs per server, and server manufacturers.

DRAM models. Figure 8 shows the relative percentages of predictable server failures decomposed by the DRAM model. The relative percentages of predictable server failures of A2, B2, B3, and C2 are lower than 10% for each failure type. On the other hand, A1, B1, and C1 account for high relative percentages of predictable server failures, up to 43.9% (B1), 32.9% (C1), and 48.8% (B1) for UE-driven, CE-driven, and miscellaneous failures, respectively.

We provide justifications for the large relative percentages of predictable server failures from some DRAM models by examining the number of server failures of each DRAM model. Table IV shows that A1, B1, and C1 have more server failures than the other DRAM models, indicating that a DRAM model with many server failures generally has a large fraction of predicable server failures.

Finding 6. *A DRAM model with a large number of failures has a large fraction of predictable server failures.*

Number of DIMMs per server. Figure 9 shows the relative percentages of predictable server failures decomposed by the

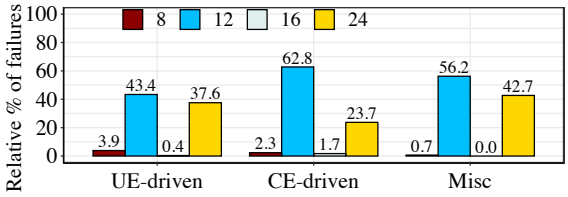


Fig. 9: Finding 7. Relative percentages of predictable server failures decomposed by the number of DIMMs per server (note that no miscellaneous failure is found for 16 DIMMs).

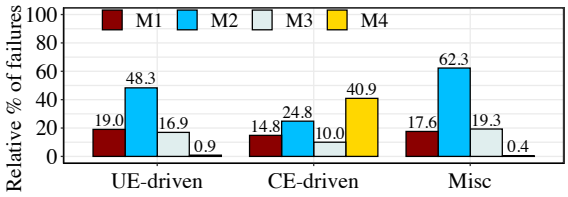


Fig. 10: Finding 8. Relative percentages of predictable server failures decomposed by the server manufacturer.

number of DIMMs per server. The predictable server failures with 12 DIMMs have the largest fraction; for example, the fraction is 43.4-62.8% for different failure types. The main reason is that the servers with 12 DIMMs have the largest population, accounting for 61.7% of all servers (Table I). On the other hand, the predictable server failures with 16 DIMMs have the lowest fraction, as the servers with 16 DIMMs due to its smallest population (2.9% of all servers). Note that the predictable server failures with 24 DIMMs have a significantly larger fraction than with 8 DIMMs, even though the servers with 8 DIMMs and 24 DIMMs have similar populations. Thus, the fraction of predictable server failures generally increases with the number of attached DIMMs.

Finding 7. *The fraction of predictable server failures generally increases with the number of DIMMs per server.*

Server manufacturers. Figure 10 shows the relative percentages of predictable server failures decomposed by the server manufacturer. The UE-driven and miscellaneous failures from M2 account for a large fraction of predictable server failures (48.3% and 62.3%, respectively), while the CE-driven failures from M4 account for the largest fraction (40.9%). The reason is that there exist large fractions of such failed servers equipped with the DRAM model B1 (i.e., 52.9% of UE-driven failures and 58.5% of miscellaneous failures from M2, as well as 48.8% of CE-driven failures from M4) (Finding 6).

Finding 8. *The fraction of predictable server failures varies across server manufacturers, yet predictable server failures are mainly attributed to the DRAM model.*

C. Characteristics of Different Types of CEs

We classify CEs into different types and analyze their characteristics. First, as CEs can be detected by memory scrubbing and memory read transactions (reported by `mcelog` (Section III)), we classify CEs into *scrubbing errors* and *read errors*, respectively. Also, we classify CEs into *soft errors* and *hard errors* based on the frequency of CE occurrences [23], [45], [46]. Specifically, if multiple CEs occur in the same cell in multiple scrubbing periods (i.e., 24 hours in our case

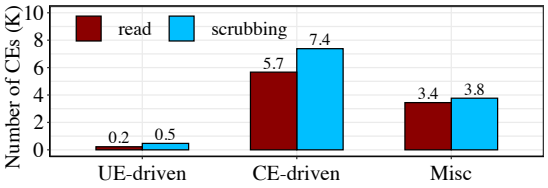


Fig. 11: Finding 9. Average numbers of scrubbing and read errors per predictable server failure.

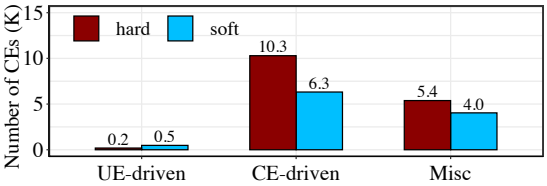


Fig. 12: Finding 10. Average numbers of hard and soft errors per predictable server failure.

(Section II)), we refer to these CEs collected from this cell as hard errors; otherwise, we refer to them as soft errors.

We first examine the average numbers of scrubbing and read errors per predictable server failure for different failure types. Figure 11 shows that the average numbers of scrubbing errors are larger than those of read errors across failure types, for example, by 1.7 K and 0.4 K for CE-driven and miscellaneous failures, respectively. It implies that memory scrubbing is critical for detecting CEs before server failures occur.

Finding 9. *Memory scrubbing generally detects more CEs than memory read transactions before server failures occur.*

We next examine the average numbers of hard and soft errors per predictable server failure for different failure types. Figure 12 shows that the average numbers of hard errors are larger than those of soft errors by 4.0 K and 1.4 K for CE-driven and miscellaneous failures, respectively. It implies that hard errors occur more commonly than soft errors for CE-driven and miscellaneous failures, conforming to the observation that hard errors are more common in prior studies [23], [43]. However, the average number of soft errors is larger than that of hard errors by 0.3 K for UE-driven failures. The reason is that system administrators put the UE-driven failed servers offline once a UE occurs, avoiding letting the soft errors on these failed servers become hard errors.

Finding 10. *Hard errors generally occur more commonly than soft errors before server failures.*

VI. Server Failure Prediction

In this section, we design a server failure prediction workflow based on our characterization of server failures (Section IV) and analysis of impacting factors on server failures (Section V).

A. Prediction Methodology

Formulation. We formulate the server failure prediction problem as an offline classification problem. Specifically, we extract and generate *features* for each server from the DRAM error logs and inventory logs, and refer to the features as input variables. We view the status of a server from the trouble tickets as a target variable. We feed the input variables into a prediction model and predict the status of a server in the near future. As our trouble tickets include three types of server failures (i.e., UE-driven, CE-driven, and miscellaneous failures) with various characteristics, we regard the server failure prediction for each failure type as a binary classification, instead of a multi-class classification for all failure types.

Feature generation. We define the *feature window* as a time interval before the prediction is conducted. During the feature window, we extract and generate features for each server with CEs (including both failed and healthy servers). As CEs occur within a short time (Section IV-C), we set the default feature window size as five minutes to include the latest CEs of servers. The number of feature windows for each server depends on its collection period within the training data (i.e., the collection period divided by the feature window size), since some servers are added into production in the midst of the dataset. Within the feature window, we extract four *feature groups*, with a total of 52 features, from each server:

- *Counter features.* We consider six counter features of CEs, including the numbers of CEs, hard errors, soft errors, read errors, and scrubbing errors, as well as the MTBE.
- *Component features.* We consider 13 component features related to component failures in two categories. First, we count the *number of CEs* from each of the seven component failures (i.e., socket, channel, bank, row, column, and cell failures, as well as random errors). Second, we count the *number of components with CEs* from each of the six component failures excluding random errors (i.e., socket, channel, bank, row, column, and cell failures).
- *Statistical features.* We calculate three statistical values (i.e., mean, median, and standard deviation) for the number of CEs from each of the six component failures excluding random errors (i.e., socket, channel, bank, row, column, and cell failures) over all the failed components in the feature window. We consider 18 statistical features in total.
- *Configuration features.* We convert the categorical values of DRAM configurations (i.e., DRAM models, number of DIMMs per server, and server manufacturers) in the inventory log into numerical values by one-hot encoding [41]. We generate 15 configuration features for different configuration settings, including seven features from the DRAM models, four features from the number of DIMMs per server, and four features from the server manufacturers.

For each server, we generate a *sample* as a collection of features within the feature window at each prediction time. Note that if there is no CE within a feature window for a server, we do not generate any sample for the server. When we perform prediction for a failure type, we refer to the samples of server failures for the failure type as positive samples, and refer to the samples of healthy servers and server failures from the other two failure types as negative samples.

Training and prediction. Before we train the prediction model, we need to address the *data imbalance issue*, i.e., the number of positive samples is much less than that of negative samples (Section III). We mitigate this issue by three steps. First, for each failure type, we train a prediction model without including the other two failure types to avoid their interference (i.e., three

prediction models in total are trained). Second, we label extra samples within one day before each server failure as positive samples (note that labeling more extra samples (e.g, within seven days) before each server failure as positive samples does not make significant differences). Finally, we downsample the negative samples [21] to make the ratio of positive to negative samples as 1:50. We have also tested different ratios (from 1:10 to 1:200), and 1:50 achieves the highest prediction accuracy.

We feed the processed samples into each of the three prediction models. We use Random Forests [7] as the default prediction model, as the previous studies [6], [35] show that Random Forests have high accuracies in disk failure [35] and DRAM error [6] prediction; we also compare different prediction models in Exp#2. We periodically predict server failures for each failure type within one day. We set the length of the time interval between two adjacent predictions (called the *prediction interval*). Currently, we set the default prediction interval as five minutes, since this is the minimum duration to relocate application services from a predicted failed server to another healthy server in production, according to the system administrators of the production data centers at Alibaba.

Evaluation metrics. To evaluate the effectiveness of server failure prediction, we adopt the time-series cross-validation by performing training and testing over the eight-month span in our dataset. Specifically, we train the prediction model in a five-month period and predict the server failures in the upcoming testing month, in which we perform prediction periodically at regular prediction intervals. We consider three sets of training (i.e., 1st to 5th, 2nd to 6th, and 3rd to 7th months) and testing periods (i.e., 6th, 7th, and 8th month, respectively). We evaluate the following accuracy metrics:

- *Precision:* The fraction of the number of server failures being correctly predicted over all server failures (including those being correctly and falsely predicted) for a failure type;
- *Recall:* The fraction of the number of server failures being correctly predicted over all actual predictable server failures (including those being correctly predicted and those being missed) for a failure type; and
- *F1-score:* $2 \times precision \times recall/(precision + recall)$.

We fix the threshold of false positive rate as 1% by adjusting the decision threshold of server failures and hyper-parameters of prediction models, which is also used in disk failure prediction [20], [50] for large-scale data centers. We report the average results for the three testing months and include the error bars that show the minimum and maximum results across the three testing months. In practice, we generate features and train prediction models only once per month, and the process of feature generation and model training takes only a few hours in total. On the other hand, we run prediction every five minutes, and the prediction process takes less than one minute for a one-month test set.

Implementation details. We implement our server failure prediction workflow in Python (v3.6.9) in 950 LoCs. Specifically, we implement downsampling using imbalanced-learn (v0.8.1) [28] and different prediction models using scikit-learn (v0.24.2) [40] and LightGBM (v3.3.1) [34].

B. Results

Exp#1 (Effectiveness of feature generation). We examine the effectiveness of feature generation for different combinations of feature groups. Figure 13 shows the precision, recall, and F1-score of different combinations of feature groups for different failure types. Using all feature groups generally increases the precision, recall, and F1-score for different failure types. For example, compared with using the counter features only, the precision using all feature groups increases from 7.7%, 35.3%, and 19.6% to 13.0%, 59.6%, and 33.4% for UE-driven, CE-driven, and miscellaneous failures, respectively, while the recall using all feature groups increases from 16.3%, 21.9%, and 27.0% to 22.5%, 66.8%, and 38.2% for UE-driven, CE-driven, and miscellaneous failures, respectively. In particular, the configuration features significantly improve both the precision and recall of CE-driven failures, implying that the configuration features generated from the categorical values can complement the other feature groups with numerical features.

However, for UE-driven and miscellaneous failures, using all features increases the precision, but decreases the recall. In particular, the highest F1-score of UE-driven failure is from using the counter, component, and statistical features, while the highest F1-score of miscellaneous failures is from using the counter and component features. A possible reason is that M2 accounts for the largest fraction (51.9%) of servers (including healthy and failed servers) among all manufacturers (Table I), while a large fraction of predictable UE-driven and miscellaneous failures are from M2 (Figure 10). Thus, the configuration features are less effective to distinguish UE-driven and miscellaneous failures from healthy servers.

Finding 11. *Using all feature groups generally increases the F1-score. The increase is significant for CE-driven failures.*

Exp#2 (Effectiveness of prediction models). We examine several classical machine learning algorithms, including logistic regression (LR) [22], support vector machine (SVM) [38], LightGBM [24], Random Forests (RF) [7], as well as a classical deep neural network algorithm Multi-Layer Perceptron (MLP) [2]. For each prediction model, we tune the hyper-parameters and decision thresholds by grid search to maximize the F1-score. In particular, for MLP, we tune the number of hidden layers from 1 to 20 and the number of neurons in each hidden layer from 10 to 200.

Figure 14 shows the precision, recall, and F1-score of various prediction models. Due to the trade-off between the precision and recall, we focus on the F1-scores of different prediction models for fair comparisons. For UE-driven failures, different prediction models have small differences on the F1-score; LightGBM achieves the highest F1-score (17.5%). For CE-driven and miscellaneous failures, tree-based prediction models (i.e., LightGBM and RF) and MLP achieve much higher F1-scores than LR and SVM. RF (LightGBM) achieves the F1-score of 62.9% (60.1%), and 35.2% (27.3%) for CE-driven and miscellaneous failures, respectively. The F1-score of MLP is lower than that of RF by up to 8.6%. A possible reason is that MLP (like general deep learning models) requires much training data for high accurate prediction, while the number of

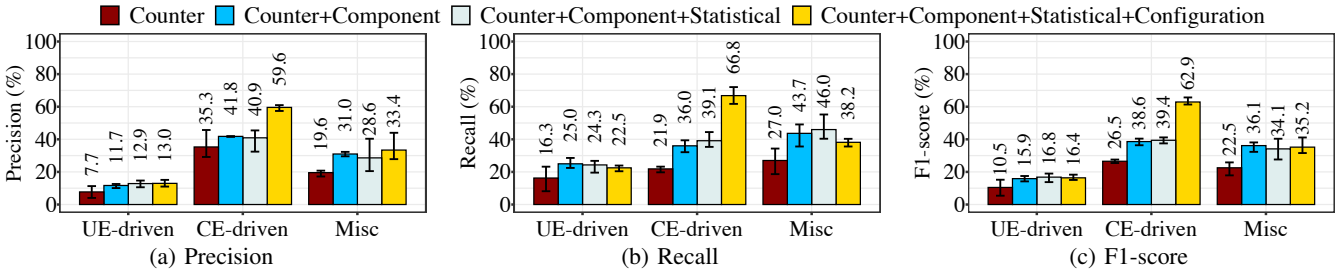


Fig. 13: Finding 11. Exp#1 (Effectiveness of feature generation).

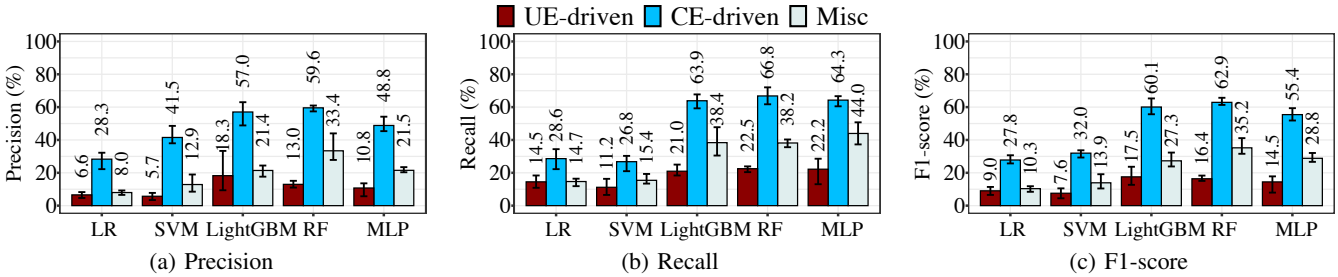


Fig. 14: Finding 12. Exp#2 (Effectiveness of prediction models).

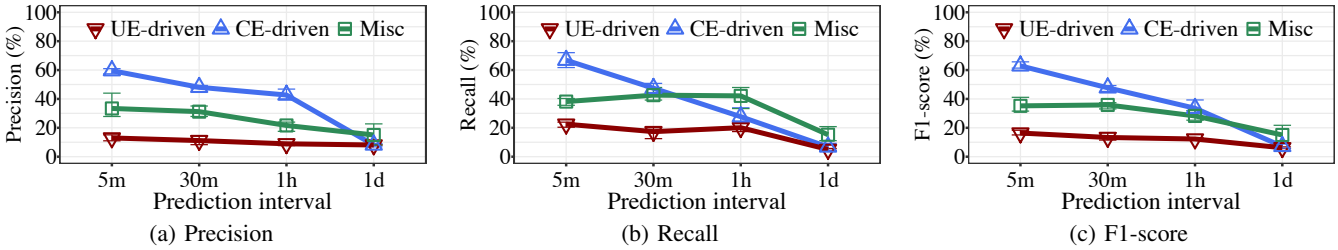


Fig. 15: Finding 13. Exp#3 (Impact of the prediction interval).

samples (including 6 K positive ones at most) for each training set after down-sampling is around 240 K obtained from 12 K servers. Also, RF is shown to achieve high accuracy in prior failure prediction studies [6], [35]. Furthermore, we observe that the F1-score of CE-driven failures is much higher than those of UE-driven and miscellaneous failures. The reason is that CE-driven failures show the symptoms of a large number of CEs before server failures happen (Finding 3), which provides more information for classification.

Finding 12. *Tree-based prediction models generally achieve the highest F1-score in our server failure prediction. Also, the prediction of CE-driven failures has the highest F1-score among the three failure types since a large number of CEs manifest before the CE-driven failures happen.*

Exp#3 (Impact of the prediction interval). Figure 15 shows the precision, recall, and F1-score for different failure types versus the prediction interval. Both the precision and recall generally decrease when the prediction interval increases. Specifically, the precision decreases to 8.1%, 8.1%, and 15.0% for UE-driven, CE-driven, and miscellaneous failures, respectively, while the recall decreases to 5.0%, 6.7%, 15.2% for UE-driven, CE-driven, and miscellaneous failures, respectively, when the prediction interval increases to one day. The reason is that the relative percentage of predictable server failures decreases when the prediction window increases (Finding 2). As a result, the F1-score of UE-driven, CE-driven, and miscellaneous failures decreases to 7.8%, 7.2%, and 15.0%, respectively, when the prediction interval increases to one day.

Finding 13. *The F1-score decreases significantly when the prediction interval increases.*

Exp#4 (Post-analysis of correctly predicted server failures). Finally, we analyze whether there is sufficient time to successfully repair the correctly predicted server failures in production. Recall that system administrators require at least five minutes to repair a server failure (Section VI-A). In our default setting, our goal is to predict server failures within one day. If a correctly predicted server failure happens at least five minutes later (and within one day), it can be successfully repaired; otherwise, it cannot be repaired. Table V shows the numbers of correctly predicted server failures that happen within next five minutes or happen at least five minutes later and within one day, as well as the total amount of reduced server downtime of all correctly predicted and successfully repaired server failures for each failure type; here, the server downtime is defined as the duration from the actual server failures until the failures are repaired, similar to the node hours due to UEs [6] and unavailable time of failed servers [30]. The numbers of UE-driven, CE-driven, and miscellaneous failures that can be successfully repaired are 27, 156, and 69 (i.e., 79.4%, 85.2%, and 98.6% among the correctly predicted server failures for the failure type), respectively. Also, predicting such correctly predicted server failures can reduce a significant amount of server downtime,

	Within 5 minutes	At least 5 minutes later	Reduced server downtime (hours)
UE-driven	7	27	1,775
CE-driven	27	156	15,520
Miscellaneous	1	69	5,975

TABLE V: Finding 14. Exp#4 (Post-analysis of correctly predicted server failures).

for example, by up to 15,520 hours for all successfully repaired CE-driven failures. Thus, our server failure prediction workflow allows a large fraction of correctly predicted server failures to be successfully repaired, and further reduces thousands of hours of server downtime.

Finding 14. *A large fraction of correctly predicted server failures can be successfully repaired, which can reduce thousands of hours of server downtime.*

VII. Related Work

Measurement of DRAM reliability. Prior studies [4], [18], [36], [37], [52] analyze DRAM errors in laboratory settings and show how DRAM errors are caused by external factors (e.g., alpha particles [36], cosmic rays [52], and ground-level radiation [4]) or internal factors (e.g., denser semiconductor technologies at lower voltages [37] and latent inherent faults in DRAM cells [18]).

Recent field studies characterize DRAM errors in production commodity data centers [23], [31], [32], [39], [43], [44]. Li et al. [31], [32] study a cluster of 212 servers, and observe that hard errors account for a non-trivial fraction among DRAM errors [31] and may lead to incorrect executions in software systems and applications [32]. Schroeder et al. [43] show that hard errors dominate among all DRAM errors in the server fleet at Google, and there exist strong temporal and spatial correlations among DRAM errors. Hwang et al. [23] measure DRAM errors and study their implications on system designs (e.g., page retirement policies and background memory scrubbing). Siddiqua et al. [44] find that CEs are mainly caused by the failures in memory controllers, buses, and channels. Meza et al. [39] characterize DRAM errors over 14 months in a server fleet at Facebook. Both [39] and ours analyze the component failures in the memory subsystem and measure the server failure rates caused by DRAM errors. However, our analysis addresses the following issues that are not considered in [39]: (i) considering the server failures that cannot normally support host applications (instead of the servers with DRAM component failures); (ii) showing how the characteristics of DRAM errors affect our subsequent predictive analysis at different time intervals before server failures happen, especially on short time scales in minutes or days (instead of on a monthly basis); and (iii) analyzing the predictability of server failures.

Several studies characterize DRAM errors in production supercomputing clusters [5], [19], [29], [45]–[47], including Jaguar [19], [46], [47], Cielo [29], [45], [47], Hopper [45], Blake [16], and an in-house supercomputing cluster containing low-power memory without ECC protection [5]. Similar to commodity data centers, hard errors dominate among all DRAM errors [46], yet the studies also draw different findings on how DRAM errors are affected by DRAM aging and the physical DRAM locations [47].

Prediction of DRAM reliability. Several studies analyze the predictability of DRAM reliability, including UEs [6], [15], [17] and failures in micro-level components (e.g., rows, columns, and cells) [13]. Giurgiu et al. [17] predict UEs based on CEs and sensor metrics (e.g., processor temperature and DRAM power). Du et al. [13], [15] use online learning to predict micro-level component failures [13], and further leverage CEs in the micro-level components to predict UEs [15]. Boixaderas et al. [6] predict UEs for supercomputers and focus on the cost-benefit analysis by quantifying the costs of UE prediction and mitigation. Yu et al. [51] take DRAM failure prediction as a multi-class classification problem by extracting handcrafted features from system kernel logs and DRAM error logs. Du et al. [14] predict whether a row fault is prone to UEs based on CE patterns and make the pages impacted by UEs offline. Wang et al. [49] leverage workload features from node-level performance metrics and cell-level DRAM access patterns for predicting UEs. In contrast, our work targets the accurate prediction of DRAM-related server failures collected from trouble tickets based on the standard accuracy metrics.

VIII. Conclusion

We present an in-depth data-driven analysis on the correlation between DRAM errors and server failures based on a large-scale dataset collected from 250 K production servers at Alibaba over eight months. Our study provides guidelines for the design of a machine-learning-based server failure prediction workflow with high prediction accuracy based on DRAM error characterization.

We highlight the findings in our server failure prediction:

- We consider three failure types in server failure prediction and train the prediction model for each failure type, instead of only UEs as in prior work [6], [15], [17]. Our prediction also reduces thousands of hours of server downtime.
- UE-driven failures are less predictable than CE-driven failures and miscellaneous failures. We achieve the F1-score up to 62.9% and 36.1% for CE-driven and miscellaneous failures. However, UE-driven failures are hard to predict, since only a small number of CEs occur before these failures (Finding 3).
- Using all feature groups increases the prediction accuracy for predicting CE-driven failures, while for UE-driven and miscellaneous failures, counter and component features are more useful than statistical and configuration features.
- The prediction interval needs to be short to predict server failures, since CEs typically occur within a short time before the prediction is conducted (Finding 2). Also, tree-based prediction models generally achieve the highest F1-score.

Acknowledgements. We thank our shepherd, Sameh Elnikety, and the anonymous reviewers for their comments. This work was supported in part by Alibaba Group via the Alibaba Innovation Research (AIR) program and the Research Matching Grant Scheme. The corresponding author is Shujie Han (shujiehan@pku.edu.cn).

REFERENCES

[1] Intel® E7500 chipset MCH Intel® x4 single device data correction (x4 SDDC) implementation and validation. https://www.intel.com.bo/content/dam/doc/application-note/e7500-chipset-mch-x4-single-device-data-correction-note.pdf.

[2] M. Anthony, P. L. Bartlett, and P. L. Bartlett. *Neural network learning: Theoretical foundations*, volume 9. 1999.

[3] M. Awasthi, M. Shevgoor, S. Kshitij, B. Rajendran, R. Balasubramonian, and V. Srinivasan. Efficient scrub mechanisms for error-prone emerging memories. In *Proc. of IEEE HPCA*, pages 1–12, 2012.

[4] R. Baumann. Soft errors in advanced computer systems. *IEEE Design and Test of Computers*, 22(3):258–266, 2005.

[5] L. Bautista-Gomez, F. Zyulkyarov, O. Unsal, and S. McIntosh-Smith. Unprotected computing: A large-scale study of DRAM raw error rate on a supercomputer. In *Proc. of ACM/IEEE SC*, pages 645–655, 2016.

[6] I. Boixaderas, D. Zivanovic, S. Moré, J. Bartolome, D. Vicente, M. Casas, P. M. Carpenter, P. Radojković, and E. Ayguadé. Cost-aware prediction of uncorrected DRAM errors in the field. In *Proc. of IEEE SC*, 2020.

[7] L. Breiman. *Random Forests*. 2001.

[8] K. K. Chang, A. Kashyap, H. Hassan, S. Ghose, K. Hsieh, D. Lee, T. Li, G. Pekhimenko, S. Khan, and O. Mutlu. Understanding latency variation in modern DRAM chips: Experimental characterization, analysis, and optimization. In *Proc. of ACM SIGMETRICS*, pages 323–336, 2016.

[9] C.-L. Chen. Error-correcting codes for byte-organized memory systems. *IEEE transactions on information theory*, 32(2):181–185, 1986.

[10] C.-L. Chen and M. Hsiao. Error-correcting codes for semiconductor memory applications: A state-of-the-art review. *IBM Journal of Research and development*, 28(2):124–134, 1984.

[11] T. J. Dell. A white paper on the benefits of chipkill-correct ECC for PC server main memory. *IBM Microelectronics Division*, 11:1–23, 1997.

[12] H. D. Dixit, S. Pendharkar, M. Beadon, C. Mason, T. Chakravarthy, B. Muthiah, and S. Sankar. Silent data corruptions at scale. *arXiv preprint arXiv:2102.11245*, 2021.

[13] X. Du and C. Li. Memory failure prediction using online learning. In *Proc. of MemSys*, pages 38–49, 2018.

[14] X. Du, C. Li, S. Zhou, X. Liu, X. Xu, T. Wang, and S. Ge. Fault-aware prediction-guided page offlining for uncorrectable memory error prevention. In *Proc. of IEEE ICCD*, 2021.

[15] X. Du, C. Li, S. Zhou, M. Ye, and J. Li. Predicting uncorrectable memory errors for proactive replacement: An empirical study on large-scale field data. In *Proc. of IEEE EDCC*, pages 41–46, 2020.

[16] K. B. Ferreira, S. Levy, V. Kuhns, N. DeBardeleben, and S. Blanchard. Understanding the effects of DRAM correctable error logging at scale. In *Proc. of IEEE CLUSTER*, pages 421–432, 2021.

[17] I. Giurgiu, J. Szabo, D. Wiesmann, and J. Bird. Predicting DRAM reliability in the field with machine learning. In *Proc. of ACM/IFIP/USENIX Middleware*, pages 15–21, 2017.

[18] S.-L. Gong, J. Kim, and M. Erez. DRAM scaling error evaluation model using various retention time. In *Proc. of IEEE/IFIP DSN*, pages 177–183, 2017.

[19] S. Gupta, T. Patel, C. Engelmann, and D. Tiwari. Failures in large scale systems: Long-term measurement, analysis, and implications. In *Proc. of IEEE SC*, pages 1–12, 2017.

[20] S. Han, P. P. Lee, Z. Shen, C. He, Y. Liu, and T. Huang. Toward adaptive disk failure prediction via stream mining. In *Proc. of IEEE ICDCS*, 2020.

[21] H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Trans. on Knowledge and Data Engineering*, 21(9), 2009.

[22] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant. *Applied logistic regression*. 2013.

[23] A. A. Hwang, I. A. Stefanovici, and B. Schroeder. Cosmic rays don't strike twice: Understanding the nature of DRAM errors and the implications for system design. In *Proc. of ACM ASPLOS*, pages 111–122, 2012.

[24] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Proc. of NIPS*, 2017.

[25] Y. Kim, R. Daly, J. Kim, C. Fallin, J. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *Proc. of ACM/IEEE ISCA*, pages 361–372, 2014.

[26] A. Kleen. Machine check handling on linux. *SUSE Labs*, 7, 2004.

[27] A. Kleen. Mcelog: Memory error handling in user space. In *International Linux System Technology Conference*, 2010.

[28] G. Lemaître, F. Nogueira, and C. K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

[29] S. Levy, K. B. Ferreira, N. DeBardeleben, T. Siddiqua, V. Sridharan, and E. Baseman. Lessons learned from memory errors observed over the lifetime of cielo. In *Proc. of ACM/IEEE SC*, pages 554–565, 2018.

[30] R. Li, Z. Cheng, P. P. Lee, P. Wang, Y. Qiang, L. Lan, C. He, J. Lu, M. Wang, and X. Ding. Automated intelligent healing in cloud-scale data centers. In *Proc. of IEEE SRDS*, 2021.

[31] X. Li, M. C. Huang, K. Shen, and L. Chu. An empirical study of memory hardware errors in a server farm. In *Proc. of USENIX HotDep*, 2007.

[32] X. Li, M. C. Huang, K. Shen, and L. Chu. A realistic evaluation of memory hardware errors and software system susceptibility. In *Proc. of USENIX ATC*, pages 75–88, 2010.

[33] X. Li, K. Shen, M. C. Huang, and L. Chu. A memory soft error measurement on production systems. In *Proc. of USENIX ATC*, pages 275–280, 2007.

[34] lightgbm. https://pypi.org/project/lightgbm/.

[35] F. Mahdisoltani, I. Stefanovici, and B. Schroeder. Proactive error prediction to improve storage system reliability. In *Proc. of USENIX ATC*, 2017.

[36] T. C. May and M. H. Woods. Alpha-particle-induced soft errors in dynamic memories. *IEEE Trans. on Electron Devices*, 26(1):2–9, 1979.

[37] A. Messer, P. Bernadat, G. Fu, D. Chen, Z. Dimitrijevic, D. Lie, D. D. Mannaru, A. Riska, and D. Milojicic. Susceptibility of commodity systems and software to memory soft errors. *IEEE Trans. on Computers*, 53(12):1557–1568, 2004.

[38] D. Meyer, F. Leisch, and K. Hornik. The support vector machine under test. *Neurocomputing*, 55(1-2):169–186, 2003.

[39] J. Meza, Q. Wu, S. Kumar, and O. Mutlu. Revisiting memory errors in large-scale production data centers: Analysis and modeling of new trends from the field. In *Proc. of IEEE/IFIP DSN*, pages 415–426, 2015.

[40] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[41] Python One-hot Encoder. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html.

[42] B. Schroeder and G. A. Gibson. A large-scale study of failures in high-performance computing systems. *IEEE Trans. on Dependable and Secure Computing*, 7(4):337–350, 2009.

[43] B. Schroeder, E. Pinheiro, and W. D. Weber. DRAM errors in the wild: A large-scale field study. *ACM SIGMETRICS Performance Evaluation Review*, 37(1):193–204, 2009.

[44] T. Siddiqua, A. E. Papathanasiou, A. Biswas, and S. Gurumurthi. Analysis and modeling of memory errors from large-scale field data collection. In *Proc. of SELSE*, 2013.

[45] V. Sridharan, N. DeBardeleben, S. Blanchard, K. B. Ferreira, J. Stearley, J. Shalf, and S. Gurumurthi. Memory errors in modern systems: The good, the bad, and the ugly. In *Proc. of ACM ASPLOS*, pages 297–310, 2015.

[46] V. Sridharan and D. Liberty. A study of DRAM failures in the field. In *Proc. of ACM/IEEE SC*, pages 1–11, 2012.

[47] V. Sridharan, J. Stearley, N. DeBardeleben, S. Blanchard, and S. Gurumurthi. Feng shui of supercomputer memory positional effects in DRAM and SRAM faults. In *Proc. of ACM/IEEE SC*, pages 1–11, 2013.

[48] G. Wang, L. Zhang, and W. Xu. What can we learn from four years of data center hardware failures? In *Proc. of IEEE/IFIP DSN*, 2017.

[49] X. Wang, Y. Li, Y. Chen, S. Wang, Y. Du, C. He, Y. Zhang, P. Chen, X. Li, W. Song, et al. On workload-aware DRAM failure prediction in large-scale data centers. In *Proc. of IEEE VTS*, 2021.

[50] J. Xiao, Z. Xiong, S. Wu, Y. Yi, H. Jin, and K. Hu. Disk failure prediction in data centers via online learning. In *Proc. of ACM ICPP*, 2018.

[51] F. Yu, H. Xu, S. Jian, C. Huang, Y. Wang, and Z. Wu. DRAM failure prediction in large-scale data centers. In *Proc. of IEEE JCC*, 2021.

[52] J. F. Ziegler and W. A. Lanford. Effect of cosmic rays on computer memories. *Science*, 206(4420):776–788, 1979.