# FACTO Dataset: A Dataset of User Reports for Faulty Computer Components

**Maria de Lourdes M. Silva, André L. C. Mendonça, Eduardo R. D. Neto, Iago C. Chaves, Carlos Caminha, Felipe T. Brito, Victor A. E. Farias, Javam C. Machado**

Laboratório de Sistemas e Banco de Dados (LSBD)
Departamento de Computação / UFC – Fortaleza – CE – Brazil

{malu.maia, andre.luis, eduardo.rodrigues, iago.chaves, carlos.caminha, felipe.timbo, victor.farias, javam.machado}
@lsbd.ufc.br

***Abstract.*** *Advancements in electronic fabrication technologies have facilitated the large-scale production of computer components, which are prone to faults over time. Despite the availability of fault-reporting tools provided by hardware manufacturers, there is a significant gap in effectively utilizing textual reports due to data scarcity. In this paper, we introduce FACTO dataset, a comprehensive collection of user reports on faulty computer components such as video cards, storage devices, motherboards, memory, and others. Data was gathered through a survey of hardware specialists, web scraping of internet forums, and synthetic text generation from real manufacturer data using large language models. This dataset aims to provide insights for correlating user reports with faulty components, thus enhancing diagnostic capabilities and improving hardware reliability and customer satisfaction.*

## 1. Introduction

The advancements in electronic fabrication technologies have led to a large-scale production of computer components. However, it is common for these components to present faults over time [Chaves et al. 2016, Queiroz et al. 2016, Lima et al. 2018]. This fact has significantly increased the need for robust diagnostic and performance assessment systems to ensure the reliability and efficiency of these devices. Hardware manufacturers have responded to this demand by providing users with tools to report faults [Park et al. 2020]. Despite these advancements, there remains a significant gap in the ability of fault-reporting platforms to effectively utilize textual reports provided by users, primarily due to data scarcity [Rombach 2023, Hakami 2024].

Textual user reports data related to computer components can be utilized in several tasks, such as fault prediction for automated customer service, quality improvement of fault-reporting platforms, sentiment analysis for marketing, and warranty policy adjustments. To effectively perform these tasks, it is essential to have a comprehensive textual dataset of user reports about faulty computer components.

In the literature, some works focus on datasets for defects and faults in computing systems [Schroeder and Gibson 2009, Xia et al. 2021]. We will now identify the works that closely resemble ours. The work [Abbas and Malik 2023] crawled user reviews from `amazon.com`. They crawled the 39,523 reviews from the top-100 of four categories: laptops, desktop computers, tablets, and computer accessories. To study the correlation between DRAM errors and server failures in data centers, the authors in [Cheng et al. 2022] gathered a dataset with 75.1 M DRAM errors from the logs of the DRAMs over an eight-month span from 250 K servers from the Alibaba data center. Also, for the goal of failure prediction for solid-state drives (SSDs), [Xu et al. 2021] collected SMART (Self-Monitoring Analysis and Reporting Technology) logs from 500 K SSDs at the Alibaba data center. There are several platforms that contain a list of datasets of computer failure, such as Usenix[1], iotta repository[2], and blackblaze[3]. However, to the best of our knowledge, none of the datasets in the literature contains textual user reports about faults in computing components.

In this paper, we introduce FACTO (FAulty CompuTer cOmponents), a dataset comprising textual reports for faulty computer components such as video cards, storage devices, motherboards, memory, and others gathered from internet forums, specialists, and a large computer manufacturer data. We provide a dataset with three key features: the textual report describing the fault, the associated computer component, and the source of the data. The data is publicly available on `https://github.com/malu-maia/FACTO`. The detailed descriptions of these features and the data collection methodology are provided in the following sections.

## 2. FACTO Dataset

In this section, we outline the construction of the FACTO Dataset, which includes English texts of user reports detailing symptoms of failures in their personal computers. For instance, a user may report to an internet forum or to the manufacturer customer service the following complaint: *"I am getting a blue screen every five minutes"*. We capture this user report from distinct sources, which adds volume and diversity to our dataset.

The dataset should be useful for correlating the user report text with the computer component that caused the failure. For that, the user report texts are labeled with the following faulty components: video card, storage, motherboard, battery, audio, CPU/FAN/Heatsink, memory, and network. We obtain the user report texts from the following sources: (i) survey where we asked for specialists to generate recurrent user reports and its solution (Section 2.1); (ii) web scrapping of internet forums; (iii) LLM-generated synthetic text data based on a non-textual report from technicians of a computer manufacturer. We describe each pipeline in detail in the next subsections. The output of each pipeline composes the full dataset.

### 2.1. Survey Data Pipeline

Hardware specialists have extensive knowledge of computing component faults. They often receive users' feedback for these kind of problem. Thus, we aim to apply a survey for specialists to collect the usual user reports that they receive. The survey uses a form to gather data, as detailed in Section 2.1.1, with the application and results discussed in Section 2.1.2.

[1] https://www.usenix.org/cfdr

[2] http://iotta.snia.org/

[3] https://www.backblaze.com/cloud-storage/resources/hard-drive-test-data

### 2.1.1. Survey Description

In the form, the specialist is requested to provide at least one answer related to a probable faulty component. The answers have to mimic users' questions that a users could ask a bot if they were having a problem with some computer component. This way, the bot could recommend troubleshooting the user's problem to the user so that potential component problems could be detected and fixed.

The questions adhere to the following pattern: "*Please enter a question that a user would ask a bot if he is experiencing a problem related to the* {`module`}", where `module` refers to the computer component under interest. Furthermore, if the specialist desires to contribute with more than one answer to one or more components, he can add additional answers at the end of the quiz.

### 2.1.2. Survey Application and Results

The survey was conducted in an R&D laboratory located in Brazil, which runs many projects based on hardware diagnostics. The survey remained open for a period of 5 days for all project members. During this period, 26 specialists applied to the survey, composed of the following roles: 17 developers, 6 quality analysts, 2 requirements analysts, and 1 manager. The reports resulted in 246 questions since every specialist is responsible for reporting at least one question for each of the aforementioned computer components, resulting in 208 questions. The remaining 38 questions were gathered from the optional additional questions.

## 2.2. Web Scraping Pipeline

Specialized hardware forums on the internet contain troubleshooting questions from real users who have failures in their computers. Company FAQs (Frequently Asked Questions) and user guides also provide usual questions for the users. We created a pipeline that followed the procedure illustrated in Figure 1. We selected some websites to collect reports about faulty components given by users, and we scraped these sites to extract useful information, filtering the helpful instances (Section 2.2.1) and applying data transformation (Section 2.2.2).

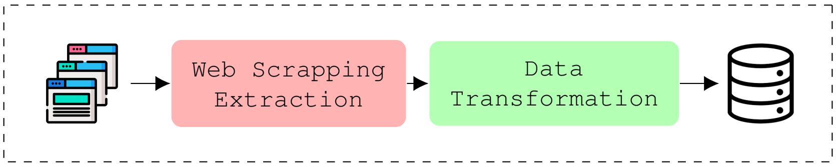


**Figure 1: Web Scrapping Pipeline.**

### 2.2.1. Websites and Data Extraction

In the data extraction stage of our dataset-building process, we searched for specialized hardware forums, company FAQs (Frequently Asked Questions), and user guides for hardware components. As primary data sources, we consider the following domains: Adrenaline[4], AMD Online Support Community[5], and NVIDIA GeForce Forums[6]. Adrenaline is one of the most active forums in Brazil. It contains a hardware section with multiple topics. The NVIDIA and AMD forum incorporates user experience and community troubleshooting regarding GPU and video card issues.

We took two approaches to this extraction: automatic web scraping for the Adrenaline forum and manual extraction from the other two sources. We used the BeautifulSoup[7] library to extract the users' reports via web scraping. We obtained 250 user reports on the first ten pages of each hardware topic of the Adrenaline forum, considering the threads labeled as "normal topics" in the platform. After scraping the three data sources, a specialized professional manually filtered the instances, resulting in 95 final instances retrieved from this process.

#### 2.2.2. Data Transformation

Some text modifications were necessary after obtaining data to transform it into user reports. The modifications are outlined below.

**Text Summarization** We observed that the text provided by the users included details about their place and sometimes a history or text containing unnecessary details and insults. A specialized professional summarized the overly detailed texts, focusing on the problem encountered by the user. The summarizing process aims to extract relevant information from texts and standardize the structure of phrases based on the previously collected instances.

**Text Expansion** Other collected instances were not complete phrases but fragments of phrases, such as "blue screen" and "no audio". Our specialized professional expanded the incomplete instances to transform them into comprehensive texts. The text expansion consists of adding stop words into the fragments to compose a complete phrase. Table 1 contains examples of text expansion provided in our data.

| Fragments of phrases | Expanded phrases |
|---|---|
| No audio. | I'm not getting any audio from my computer. |
| Not paring. | My Bluetooth device is not pairing with my computer. |
| Blue screen. | My computer is showing a blue screen with an error message. |
| Timed out. | My connection keeps timing out when I try to access the internet. |

**Table 1. Examples of expanded fragments of phrases.**

**Text Translation** As the Adrenaline forum is a Brazilian platform, the users wrote their complaints and doubts in Portuguese. Since our dataset contains only English data, the data collected from this source were translated into English to join our dataset.

### 2.3. Manufacturer's Data Pipeline

The manufacturer of a large hardware company provided a tabular dataset containing data from their repair center, with 15,453 rows and 154 columns. Each line from this

[4] Adrenaline: https://forum.adrenaline.com.br/

[5] AMD: https://community.amd.com/t5/support-forums/ct-p/supprtforums

[6] NVIDIA: https://www.nvidia.com/en-us/geforce/forums/support/

[7] Beautiful Soup library: https://beautiful-soup-4.readthedocs.io/en/latest/

dataset contains data about the technicians' diagnostic on a failure describing the observed problems of the failure, e.g., "blue screen" or "no sound", and the associated repair action, e.g., "replace motherboard".

More specifically, there is a column named problem_category related to predefined problem categories as the category "no audio" where the technician just indicates if the problem "no audio" is present or not. The column problem_category contains a list of all the categories that the technician marked as present. There is also a column named technician_symptom_details which holds a less detailed but expert-validated technical description of the problem where the technician writes a short description of the problem. Finally, the column problem_solution contains the associated repair action.

This, the failure problems are described using short sentences and abbreviations. Thus we need means to generate full sentences mimicking user reports for faulty components based on this real manufacturer dataset. For that, we apply an LMM-based approach to create synthetic text as delineated in Figure 2 and described in the following sections.

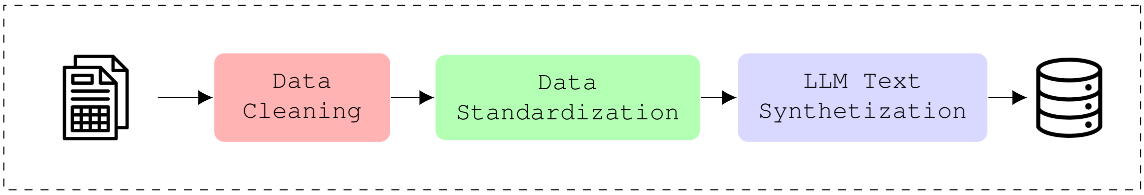


**Figure 2: Manufacturer's Data Pipeline.**

### 2.3.1. Data Cleaning and Standardization

Most of the dataset's columns were irrelevant to our aim or had redundant information. We dropped columns containing machine serial numbers, diagnostic data for multiple tests, product names, and IDs, among other unnecessary fields for our context. We selected the three relevant columns: problem_category, technician_symptom_details and problem_solution.

The faulty computer components in the manufacturer's data did not match the modules' names used to label the reports made by the users collected in the previous data extraction. We standardized the names of the faulty components to match the labels used in the previous text classifications. A sample of the mappings is presented in Table 2a.

## 2.4. Models Evaluation

Some components, such as peripheral members, were not included in our data, so they did not match any of our labels. We dropped the unmatched rows and the duplicates, and the final manufacturer's data includes 433 rows containing a list of problem categories, the problem detailed by a technician, the problem resolution, and the faulty module.

### 2.4.1. LLM Text Synthetization

In this step, we generated synthetic text data using an Large Language Model (LLM). Our approach is based on non-textual reports from technicians, as described in the previous sections. For that, we use the categories (column problem_category) and the technician

| Manufacturer | Default |
|---|---|
| Systemboard | Motherboard |
| Speakers | Audio |
| Wireless/WWAN Card | Network |
| Hard Drive | Storage |

(a)

| Module | # Seed Instances | # Synth. Instances |
|---|---|---|
| Motherboard | 174 | 174 |
| Battery | 118 | 118 |
| Audio | 50 | 50 |
| CPU/FAN/Heatsink | 44 | 44 |
| Network | 22 | 44 |
| Storage | 21 | 42 |
| Video Card | 3 | 30 |
| Memory | 1 | 10 |
| | **433** | **512** |

(b)

**Table 2. Table (a) contains a sample of the faulty computer components and their mapped label and Table (b) contains the number of synthetic descriptions generated per module in the FACTO dataset.**

report (column technician_symptom_details) as seeds so that LLM can generate rich descriptions about these problems mimicking a real user.

Thus, for each example in the clean and standardized dataset (totaling 443 examples), we aimed to generate varied descriptions, simulating different user profiles with varying levels of technical knowledge. The number of descriptions generated for each example depended on its category due to data imbalance. For categories with more examples, only one description was generated. However, for underrepresented categories, the number of descriptions was increased to generate between 40 and 50 examples. Specifically, "Network" and "Storage" had 2 descriptions each, and "Video Card" and "Memory" had 10 descriptions each. This approach ensured a more balanced dataset. This was done using a zero-shot learning approach, as no actual descriptions were used to train the model. Table 2b illustrates the number of synthetic descriptions planned to be generated for each module of the FACTO dataset.

The prompt design emphasized the generation of varied and realistic descriptions in English, with explicit instructions to avoid Portuguese or any other language. It was structured to ensure that the model produced ten descriptions per example, formatted as a Python dictionary. The repetition of instructions within the prompt aimed to reinforce the importance of adherence to the specified format and language. Listing 1 shows the complete prompt used. The number of descriptions, the problem category, and the detailed symptoms are parameters for this.

```
    You are an intelligent agent designed to generate synthetic
descriptions of computer problems based on provided categories and
detailed symptoms.

    You will generate {num_descriptions} varied and realistic
descriptions in English, sometimes using technical language,
other times using layman language, simulating diverse user
profiles.

    It is imperative that all descriptions are generated in
English only. Under no circumstances should any descriptions be in
Portuguese or any other language.
```

```
    Problem Category: {problem_category}
    Detailed Symptom: {problem_detailed_symptom}

    Generate exactly {num_descriptions} synthetic problem
descriptions based on the above information. In the descriptions
generated, do not make reference to well-known brands of computers
or notebooks, such as Macbook, Lenovo or DELL, make references
only to computers, notebooks or tablets. Provide the descriptions
in the form of a Python dictionary with the key "descriptions" and
the value being a list of exactly {num_descriptions} descriptions.
Return only the dictionary, without any additional text before or
after it.

Response:
```

**Listing 1. Prompt for Synthetic Data Generation**

In this work, the manufacturer did not allow their data to be processed by external proprietary services like GPT, GEMINI, or CLAUDE. This, we need to use a open-source model. We employed the ctransformers library [Marella 2023] (version 0.2.27) for this task, specifically utilizing the Yi LLM [Young et al. 2024]. The 34-billion parameter version, pre-quantized to 4 bits, was chosen. This model was quantized using the GPT-Generated Unified Format (GGUF) method [Chavan et al. 2024], significantly reducing the VRAM requirements for inference. This method enables efficient inference by distributing layers between the GPU and CPU. In our setup, 55 out of the 60 layers were processed on the GPU, while the remaining layers ran on the CPU. The hardware used for this task included an RTX3090 GPU with 24 GB of VRAM, 128 GB of RAM, and an Intel I9 10850K processor with 10 cores and 20 threads. During the data generation process, the Python script consumed approximately 22 GB of VRAM and 8 GB of RAM.

In total, 512 synthetic problem descriptions were generated. These descriptions were compared with the survey data to assess their quality and similarity. Using the Sentence-BERT (S-BERT) transformer [Reimers and Gurevych 2019], we extracted 384 numerical features for each text. These high-dimensional vectors were then reduced to two dimensions using t-SNE [Van der Maaten and Hinton 2008] for visualization purposes. Figure 3 illustrates this comparison: (a) shows the survey and the scraped data, and (b) shows the synthetic data. Both visualizations exhibit some common patterns: the labels “Audio”, “Battery”, and “Network” are spatially more distinct from other labels, while “Motherboard”, “Storage”, “Memory” and “CPU/FAN/Heatsink” exhibit high spatial overlap due to common symptoms like “blue screen”, crashes, and slow performance.

This analysis indicates that the synthetic data closely mimics the structure of the survey data, providing some validation of the effectiveness of our synthetic data generation approach in creating realistic user-reported problem descriptions. This is particularly relevant because the approach used is Zero-Shot learning (without receiving descriptions of the survey and scraped data). In other words, from a list of problem and symptom categories, LLM was able to generate rich descriptions with some similarity to real data. This dataset, with its diverse range of descriptions, provides a valuable resource for correlating user reports with the corresponding faulty computer components, enhancing the

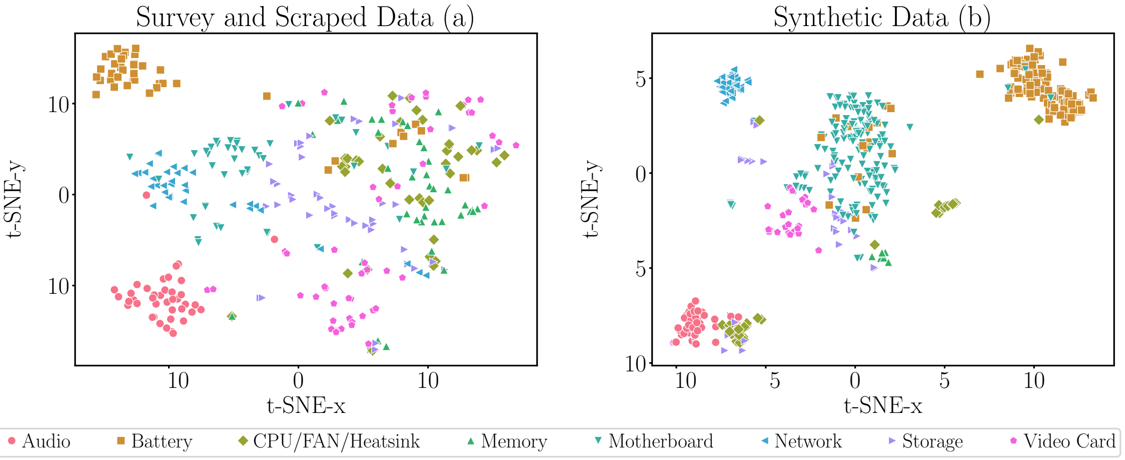


**Figure 3: Dimensionality reduction of the features extracted by the transformer. (a) Survey and Scraped Data. (b) Synthetic Data.**

diagnostic capabilities for computer failures.

### 2.5. Format and Data Content

After combining three datasets from various sources, we created the FACTO dataset, which consists of 853 instances. Of these, 246 were gathered from a specialist survey, 95 were scraped from hardware forums, and 512 were synthetic instances generated using the manufacturer's data. The dataset includes three columns. The first column contains user-reported computer failures, the second contains the faulty component associated with the report given by the user, and the third contains the source of this information. Table 3 displays the number of samples held by each module.

| Module | # Instances |
|---|---|
| Motherboard | 221 |
| Battery | 161 |
| Audio | 91 |
| Storage | 90 |
| CPU/FAN/Heatsink | 83 |
| Video Card | 81 |
| Network | 77 |
| Memory | 49 |

**Table 3. Number of reports for each module in FACTO dataset.**

## 3. Applicability

An important task that can be executed with our dataset is creating an NLP model that, given the text of a user report for hardware failure, predicts the faulty component that caused the failure. This model can be applied to automated customer services where users type their reports to the manufacturer customer contact channel and it automatically identifies the issue with their computers and suggests an action like running tests or sending their computer to the repair center.

Hardware engineers can also investigate which components are more prone to failures. Also, many messages contain details about the conditions and symptoms of the failure, which gives more input to grasp the root cause of the failure. This way, engineers can redesign components and implement new testing protocols to improve hardware durability and, consequently, customer satisfaction.

Additionally, with FACTO, one can develop NLP systems for sentiment analysis to understand how customers feel about the failures. Negative sentiments can hurt the company's reputation. This way, the marketing team can create strategies to mitigate customer dissatisfaction with communication to provide transparency and build trust. Finally, the dataset provides prior knowledge of the recurrent faulty components and the sentiment of the user reports. This allows for adjusting warranty policies to alleviate customer inconvenience from failing computers.

## 4. Explanatory Analysis

This section discusses some analysis made on the FACTO dataset. Figure 4 shows the occurrences of the reports of failure components grouped by module for each source. The purple bar represents the instances collected from the forums, the blue bar represents the reports collected by the survey, and the green bar shows the count of synthetic instances.

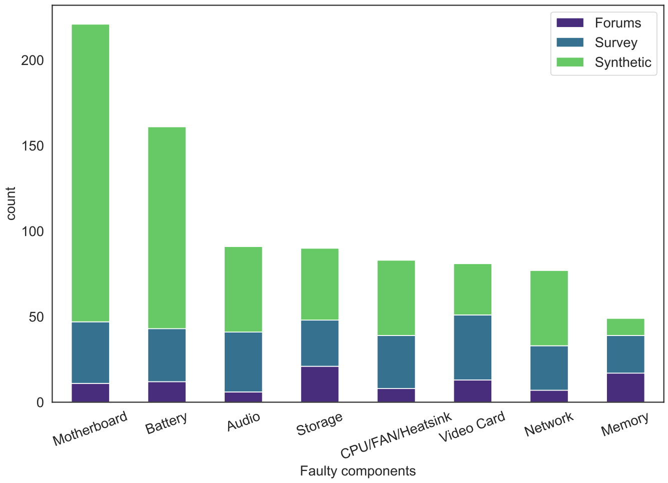
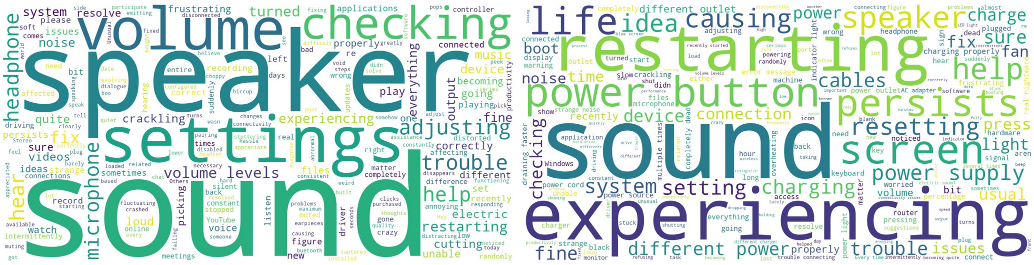


**Figure 4: Histogram of the users reports occurrences grouped by module.**

Although the synthetic instances are unbalanced, they are proportional to the manufacturer's data. Since there was a lack of reports about memory, a few instances labeled as "Memory" belong to this source. In the FACTO dataset, the "Motherboard" class contains more samples, with 221 examples, while the "Memory" has only 10 samples. We grouped the reports classified to each component to discover tips that can lead to a potentially faulty component and analyzed the corresponding wordclouds. Based on the graphics shown in Figure 5a, the most frequent words when users encountered issues in the "Audio" component were "sound", "speaker", "volume", and "settings". Those tips are very meaningful since all these words can be related to audio. Furthermore, Figure 5b contains the most used words in the FACTO dataset, where the most significant words frequently appeared in more than one component, such as "sound". Users reported hearing sounds from a particular piece of hardware, experiencing computer "beeps", or encountering audio issues. Since some terms can apply to more than one component, it's essential to consider the context.

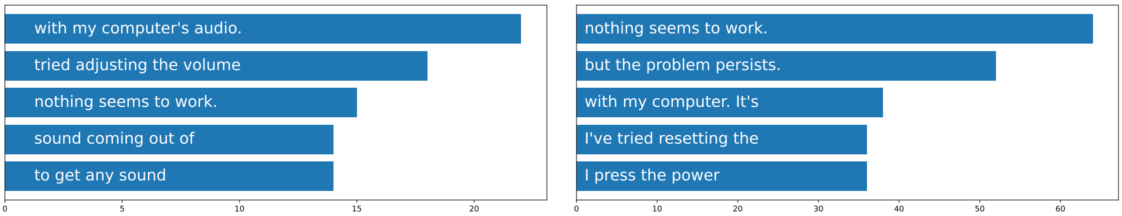


a) Audio.

b) FACTO.

**Figure 5: Figure (a) shows the wordcloud of the users' reports about the Audio component, while Figure (b) illustrates the wordcloud that represents the whole data, including all faulty components.**

The analysis of n-grams can also lead a technician or a classifier to some tips about what a report could contain when a user encounters issues in its computer component. In our example illustrated in Figures 6a and 6b, we employed 4-grams in data samples. For instance, reports labeled as audio contain 22 samples with the sentence "with my computer's audio", which may cause the technician to suppose that the problem is associated with the audio component. In general, the most frequent 4-grams in the FACTO dataset is the user complaining that "nothing seems to work" with 64 occurrences.



a) Audio.

b) FACTO.

**Figure 6: Figure (a) illustrates the 4-grams of the users' reports about the Audio component, and (b) contains the 4-grams of all users' reports in FACTO.**

## 5. Conclusion

FACTO dataset addresses a gap in the effective utilization of textual user reports for diagnosing faulty computer components. By incorporating data from surveys of hardware specialists, web scraping of internet forums, and synthetic text generation using large language models, FACTO offers a diverse and comprehensive collection of user-reported issues. The dataset enhances the capabilities of automated customer service systems, improves the quality of fault-reporting platforms, and provides valuable insights for marketing and warranty policy adjustments. Future work should expand the dataset to include a wider range of hardware components and user scenarios, involve end users in the survey, and refine the data collection and processing methodologies to further improve the quality and applicability of the dataset.

## Acknowledgment

This research was partially funded by Lenovo, as part of its R&D investment under Brazilian Informatics Law.

## References

Abbas, Y. and Malik, M. S. I. (2023). Defective products identification framework using online reviews. *Electronic Commerce Research*, 23(2):899–920.

Chavan, A., Magazine, R., Kushwaha, S., Debbah, M., and Gupta, D. (2024). Faster and lighter llms: A survey on current challenges and way forward. *arXiv preprint arXiv:2402.01799*.

Chaves, I. C., de Paula, M. R. P., Leite, L. G., Queiroz, L. P., Gomes, J. P. P., and Machado, J. C. (2016). Banhfap: A bayesian network based failure prediction approach for hard disk drives. In *2016 5th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 427–432.

Cheng, Z., Han, S., Lee, P. P., Li, X., Liu, J., and Li, Z. (2022). An in-depth correlative study between dram errors and server failures in production data centers. In *2022 41st International Symposium on Reliable Distributed Systems (SRDS)*, pages 262–272. IEEE.

Hakami, A. (2024). Strategies for overcoming data scarcity, imbalance, and feature selection challenges in machine learning models for predictive maintenance. *Scientific Reports*, 14(1):9645.

Lima, F. D. S., Pereira, F. L. F., Chaves, I. C., Gomes, J. P. P., and Machado, J. C. (2018). Evaluation of recurrent neural networks for hard disk drives failure prediction. In *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 85–90. IEEE.

Marella, R. (2023). ctransformers:python bindings for the transformer models implemented in c/c++ using ggml library. `https://github.com/marella/ctransformers`. Accessed: 2024-06-29.

Park, Y., Fan, S., and Hsu, C. (2020). A review on fault detection and process diagnostics in industrial processes. processes, 8 (9), 1123.

Queiroz, L. P., Rodrigues, F. C. M., Gomes, J. P. P., Brito, F. T., Chaves, I. C., Paula, M. R. P., Salvador, M. R., and Machado, J. C. (2016). A fault detection method for hard disk drives based on mixture of gaussians and nonparametric statistics. *IEEE Transactions on Industrial Informatics*, 13(2):542–550.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Rombach, K. (2023). *Fault Diagnostics under label and data scarcity*. PhD thesis, ETH Zurich.

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

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Xia, F., Song, H., Yan, L.-C., Li, Y., and Wang, L.-J. (2021). A survey on failure prediction in large-scale computing systems. In *2021 IEEE 23rd Int Conf on High Performance*

*Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, pages 2028–2033. IEEE.

Xu, F., Han, S., Lee, P. P., Liu, Y., He, C., and Liu, J. (2021). General feature selection for failure prediction in large-scale ssd deployment. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 263–270. IEEE.

Young, A., Chen, B., Li, C., Huang, C., Zhang, G., Zhang, G., Li, H., Zhu, J., Chen, J., Chang, J., et al. (2024). Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.