

By My Eyes: Grounding Multimodal Large Language Models with Sensor Data via Visual Prompting

Anonymous ACL submission

Abstract

Large language models (LLMs) have demonstrated exceptional abilities across various domains. However, utilizing LLMs for ubiquitous sensing applications remains challenging as existing text-prompt methods show significant performance degradation when handling long sensor data sequences. We propose a visual prompting approach for sensor data using multimodal LLMs (MLLMs). We design a visual prompt that directs MLLMs to utilize visualized sensor data alongside the target sensory task descriptions. Additionally, we introduce a visualization generator that automates the creation of optimal visualizations tailored to a given sensory task, eliminating the need for prior task-specific knowledge. We evaluated our approach on nine sensory tasks involving four sensing modalities, achieving an average of 10% higher accuracy than text-based prompts and reducing token costs by $15.8\times$. Our findings highlight the effectiveness and cost-efficiency of visual prompts with MLLMs for various sensory tasks.

1 Introduction

Large language models (LLMs) have shown remarkable performance in tasks across diverse domains, including science, mathematics, medicine, and psychology (Bubeck et al., 2023). The recent advent of multimodal LLMs (MLLMs), *e.g.*, GPT-4o (OpenAI, 2024), has further expanded their capabilities to images and audio inputs, broadening their use in fields such as industry and medical imaging (Yang et al., 2023). Meanwhile, sensor data, widely used in healthcare (Ferguson et al., 2022) and environmental monitoring (Hayat et al., 2019), hold potential for ubiquitous applications when effectively integrated with MLLMs. However, the diversity of sensors (Wang et al., 2019) and the heterogeneity among them (Stisen et al., 2015) hinder the implementation of a foundational model that generalizes across various sensing tasks.

Text-only Prompt

"Based on the data, classify whether the user is walking or running.
Given sensor data: $[[0.65, 0.62, -0.36], [0.65, 0.63, -0.37], [0.65, 0.63, -0.35], [0.65, 0.63, -0.36], [0.65, 0.63, -0.36], [0.65, 0.61, -0.37], [0.64, 0.61, -0.36], [0.64, 0.61, -0.36], \dots]$ "

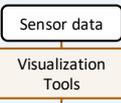
MLLM Response: "walking" ❌ 52,910 tokens

Our method

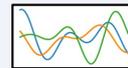
Visualization Generator

"What is the best visualization with data for classifying walking or running?"

MLLM Response: "waveform plot"



Visual Prompt



"Based on the visualized sensor data, classify whether the user is walking or running."

MLLM Response: "running" ⓪

2,020 tokens

Figure 1: An example of solving a sensory task using an MLLM with visual prompts. The visualization generator generates an appropriate visualization for the given sensor data, and the visualized data is provided as an image to the MLLM for solving the task.

The expensive data collection (Vijayan et al., 2021) often results in insufficient training data, further complicating the development of such capability.

Recent studies explored leveraging pre-trained LLMs to solve general sensory tasks (Yu et al., 2023; Liu et al., 2023; Kim et al., 2024). One approach extracts task-specific features from sensor data and composes them as prompts (Yu et al., 2023). However, designing such prompts requires specific domain knowledge. Alternatively, incorporating raw sensor data as text prompts (Kim et al., 2024; Liu et al., 2023) has been a widely used method to handle sensory data with LLMs as a more generalizable solution. Yet, we empirically found that providing raw sensor data with text prompts shows poor performance in real-world sensory tasks with long-sequence inputs and incurs

059 high costs due to an extensive number of tokens.

060 To address these challenges, we propose provid- 109
061 ing visualized sensor data as images to MLLMs 110
062 that support visual inputs. Leveraging MLLMs’ 111
063 growing ability to interpret visual aids (Yang et al., 112
064 2023), we explore their effectiveness in analyzing 113
065 plots generated from sensor data. We designed a 114
066 visual prompt comprising visualized sensor data 115
067 and task-specific instructions to solve sensory tasks. 116
068 In addition, we present a visualization generator 117
069 that enables MLLMs to independently generate 118
070 optimal visualizations using tools available in pub- 119
071 lic libraries. This generator filters potential visu- 120
072 alization methods based on the task description 121
073 and assesses the resulting visualizations of each 122
074 method to determine the best visualization. Fig- 123
075 ure 1 compares the existing text-only prompts with 124
076 our method for sensory tasks. 125

077 Evaluations on nine sensory tasks involving four 126
078 different modalities showed that visual prompts 127
079 generated from the visualization generator signifi- 128
080 cantly improved performance by an average of 10% 129
081 while reducing token costs by $15.8\times$ compared 130
082 with the existing baseline. Our findings highlight 131
083 the effectiveness and efficiency of visualized sensor 132
084 data with MLLMs in various applications. 133

085 We summarize our contributions as follows: 134

- 086 • We propose to ground MLLMs with sensor 135
087 data by providing visualized sensor data as 136
088 images, achieving improved performance at 137
089 reduced costs than the text-based baseline. 138
- 090 • We present a visualization generator that auto- 139
091 matically generates suitable visualizations for 140
092 various sensory tasks using public libraries. 141
- 093 • We conduct experiments on nine different sen- 142
094 sory tasks across four modalities, demonstrat- 143
095 ing the broad applicability of our approach. 144

096 2 Related Work 145

097 **LLMs with sensor data.** Sensory tasks involve 146
098 sequences of numbers indicating values over time. 147
099 Initial research for handling sequential data focused 148
100 on time-series forecasting (Zhang et al., 2024b). 149
101 Converting time-series data into text prompts for 150
102 forecasting has been proposed in PromptCast (Xue 151
103 and Salim, 2023) and LLMTime (Gruver et al., 152
104 2024). Other studies (Zhou et al., 2023; Jin et al., 153
105 2023a) used specialized encoders to create embed- 154
106 dings compatible with pre-trained LLMs. 155

107 Beyond forecasting, LLMs have been explored 156
108 in healthcare for their ability to answer questions 157

109 using physiological sensor data (Liu et al., 2023). 110
111 For example, LLMs have been used for ECG diag- 112
113 nosis (Yu et al., 2023) by integrating ECG-specific 114
115 features and retrieval-augmented knowledge from 116
117 ECG databases. Penetrative AI (Xu et al., 2024) 118
119 and Health-LLM (Kim et al., 2024) have used raw 120
121 sensor data in text prompts to solve health prob- 122
123 lems without task-specific processing. Our study 124
125 examines whether existing methods can general- 126
127 ize to broader sensing tasks with high-frequency, 128
129 long-duration data. Building upon these works, 130
131 we propose visualizing sensor data for MLLMs to 132
133 improve their performance and cost efficiency. 134

135 **Multimodal large language models (MLLMs).** 136
137 Advancements in MLLMs (Zhang et al., 2024a) 138
139 have equipped popular models such as Chat- 140
141 GPT (OpenAI, 2022) with vision capabilities (Open- 142
143 AI, 2024). Recent studies explored the in-context 144
145 learning (Brown et al., 2020) abilities of MLLMs, 146
147 showing that they can understand images with 148
149 the interleaved text and few-shot examples (Tsim- 149
150 poukelli et al., 2021; Alayrac et al., 2022). This 150
151 capability has been applied in medical diagnostics, in- 152
153 cluding analyzing radiology and brain images with 153
154 accompanying text instructions (Wu et al., 2023). 154
155 Our work explores using MLLMs to analyze visu- 155
156 alized sensor data for broader applications. 156

157 **Using tools with LLMs.** Recent research has 157
158 shown that augmenting LLMs with external tools 158
159 can extend their capabilities. Toolformer (Schick 159
160 et al., 2024) enables LLMs to access public 160
161 APIs and search engines, while Visual Program- 161
162 ming (Gupta and Kembhavi, 2023) uses LLMs to 162
163 generate and execute codes. HuggingGPT (Shen 163
164 et al., 2024) and Chameleon (Lu et al., 2024) in- 164
165 tegrated multiple expert models to enhance func- 165
166 tionalities. Our work builds upon these works by 166
167 enabling MLLMs to utilize sensor data visualiza- 167
168 tion tools. Moreover, we propose a design that not 168
169 only uses tools but also allows MLLMs to assess 169
170 their effectiveness, ensuring optimal visualization 170
171 for specific tasks. 171

172 3 Limitations of Representing Sensor 172 173 Data as Text-based Prompts 173

174 Existing approaches for grounding language mod- 174
175 els with sensor data primarily rely on text-based 175
176 prompts (Liu et al., 2023; Jin et al., 2023b; Zhang 176
177 et al., 2024b; Yu et al., 2023). One approach uses 177
178 prompts with specialized features extracted from 178
179 sensor data for specific tasks, such as R-R inter- 179

vals for ECG-based applications (Yu et al., 2023). While this approach effectively handles known sensory tasks, feature-based prompts often require domain knowledge, which is not generalizable for non-expert users. Instead, a more common approach (Kim et al., 2024; Xu et al., 2024; Liu et al., 2023) incorporates raw sensor data sequences directly into prompts without data processing. However, most studies focus on short sequences (e.g., fewer than 100 elements) (Kim et al., 2024) and simple tasks (e.g., binary classification) (Liu et al., 2023).

Real-world sensor data often entail long numeric sequences with high sampling rates and long durations. For example, arrhythmia detection (Wagner et al., 2020) requires ECG data sampled at 100Hz over 10 seconds, resulting in 1,000 elements. This section investigates the limitations of using text-based prompts to represent such complicated sensor data in language models. We focus on the capability to interpret sensor data and the token consumption costs associated with long numeric sequences.

Language models struggle to interpret long numeric text sequences. Language models interpret simple numeric sequences by performing arithmetic operations (Achiam et al., 2023) and understanding sequential data (Gruber et al., 2024; Mirchandani et al., 2023). However, we empirically revealed that their performance declines significantly with longer sequences inside the prompt, such as those exceeding 100 numbers, common in sensor data.

We conducted experiments with two specific tasks: *mean prediction* to evaluate arithmetic capabilities (Pirttikangas et al., 2006) and *wave classification* to assess pattern recognition (Liu et al., 2016) in sequences. The defined tasks represented the basic functionalities for sensor data interpretation, serving as typical feature extraction methods. Using randomly generated sine and sawtooth waves with varying lengths, we asked a language model, GPT-4o (OpenAI, 2024), to calculate mean values and classify wave types using one-shot examples for each task. Each task was repeated 30 times to ensure robustness.

Figure 2 shows the results. In arithmetic operations, error rates consistently increased with the length. In pattern recognition, performance declines significantly for sequences longer than 100 elements, approaching the performance of a random classifier at 500 elements. While recent

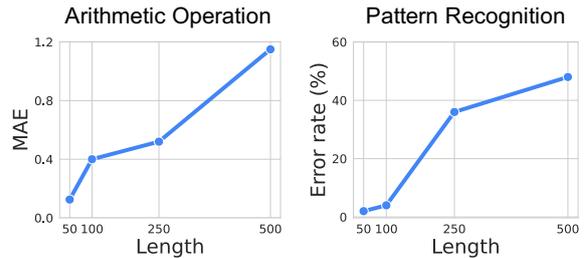


Figure 2: Performance of GPT-4o on arithmetic operation (mean prediction) and pattern recognition (sine and sawtooth wave classification) tasks for varying lengths.

models such as GPT-4o, with its 128K context window, support long input lengths, our results indicate that interpreting sensor data with long numeric sequences still remains challenging.

Sensor data in text is costly. The computational and financial burdens for API users of language models scale with the number of tokens in the prompt. Representing sensor data in textual format leads to extensive token usage, thereby increasing costs. For instance, performing passive sensing to track activities with smartphone accelerometers (Stisen et al., 2015) uses sensor data sampled at 100 Hz. Collecting this data over a minute results in a prompt of 18K numbers, translating to 90K tokens. This leads to a huge cost of \$450 per hour when using GPT-4o API to classify six activities with one example for each. Higher sampling rates or longer durations further increase the costs, making such applications infeasible.

Transition to visual prompts. Language models such as ChatGPT (GPT-4o (OpenAI, 2024)) and Gemini (DeepMind, 2024) have expanded capabilities to include multimodal inputs (e.g., vision and audio). Recent Multimodal Large Language Models (MLLMs) demonstrate an increasing ability to identify patterns and interpret visual data (Achiam et al., 2023). This opens new opportunities for sensory tasks, as sensor data are often visualized for analysis. Visualizations make complex data more interpretable and condense long data sequences into a *single* image, significantly reducing token costs. Building on this capability, we exploit visualized sensor data instead of text-based prompts.

4 Method

We introduce our method for handling sensory tasks by providing sensor data as image inputs to MLLMs. Section 4.1 overviews our prompt design strategy. Section 4.2 introduces our visualization

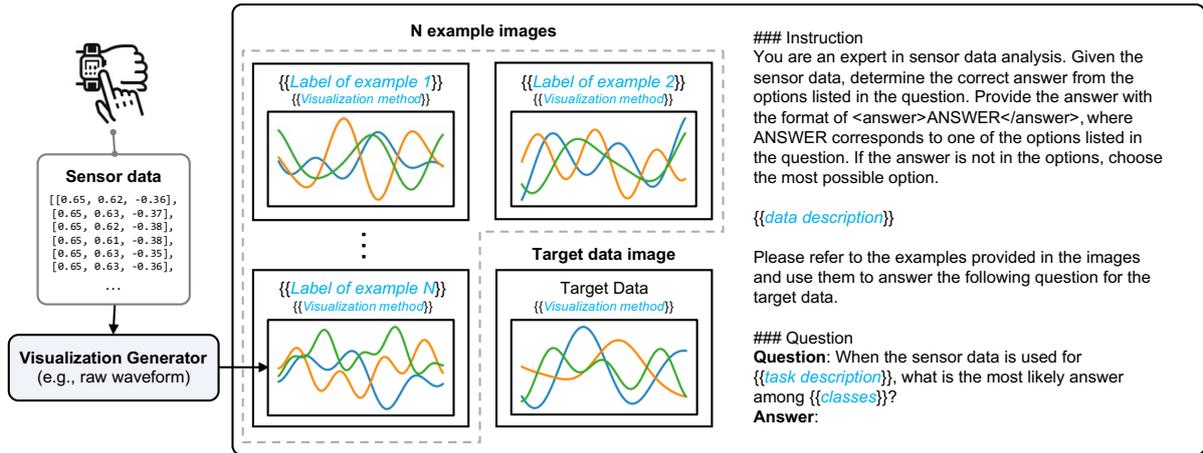


Figure 3: Overview of our visual prompt. Sensor data are transformed into annotated images with labels and visualization methods. Additionally, instructions are provided to the MLLM, detailing the task and relevant data descriptions. These instructions guide the MLLM on effectively utilizing the provided images to solve the task.

generator, which automatically generates suitable visualizations for heterogeneous sensor data.

4.1 Visual Prompt Design

To leverage MLLMs for sensory tasks, we propose a *visual prompt*, as illustrated in Figure 3. The key idea is to transform numeric sequences of sensor data into visual plots using various methods, such as raw waveforms and spectrograms. Detailed information about these visualization methods is in Section 4.2. For few-shot examples, each plot includes a label as a title above it (i.e., *Label of example X*). For unlabeled target data used in queries, the title is simply stated as “target data”. We provide textual instructions to clarify the data collection process and the task’s objectives. These instructions ensure that MLLMs can effectively interpret and utilize the visualized sensor data.

4.2 Visualization Generator

In our proposed visual prompt, the choice of visualization method is crucial, as it significantly influences the MLLM’s ability to comprehend the sensor data. For example, raw waveform plots are ideal for tasks involving amplitude pattern recognition over time, while spectrograms (Ito et al., 2018) are suitable for tasks relying on frequency features. We introduce a *visualization generator* that automatically chooses the most suitable visualization tool from available public libraries, enabling non-expert users to effectively utilize visual prompts. This generator operates in two main phases: (i) visualization tool filtering and (ii) visualization selection (see Figure 4).

Visualization tool filtering. Public libraries offer a vast array of sensor data visualizations. However, trying each out to identify the optimal visualization is computationally expensive. To minimize the cost, we employ a filtering approach. By providing available visualization tools, descriptions of the task, and data collection, we ask MLLMs to select a list of visualization methods useful for the target task.

As shown in Figure 4 (green box), we provide a full list of available visualization methods found in public libraries (e.g., Matplotlib (Hunter, 2007), Scipy (Virtanen et al., 2020), and Neurokit2 (Makowski et al., 2021)) along with task and data collection descriptions to MLLMs. We also leverage the in-context learning ability of MLLMs to enhance response quality by providing demonstrations of optimal visualizations chosen for different tasks. The MLLM is instructed to output a list in JSON format, which is suitable for automated parsing at a later stage. Appendix C shows the full list of available visualization tools and demonstrations.

Visualization selection Sensor data exhibit variations by instance due to user-specific behaviors, environmental factors, or device settings (Stisen et al., 2015), which cannot be fully captured in task and data descriptions. This variability limits the reliability of selecting visualizations based solely on the descriptions. To address this, we visualize the sensor data using all filtered visualization tools and ask the MLLM to select the one that provides the best visual information for the task.

The blue box in Figure 4 illustrates this proce-

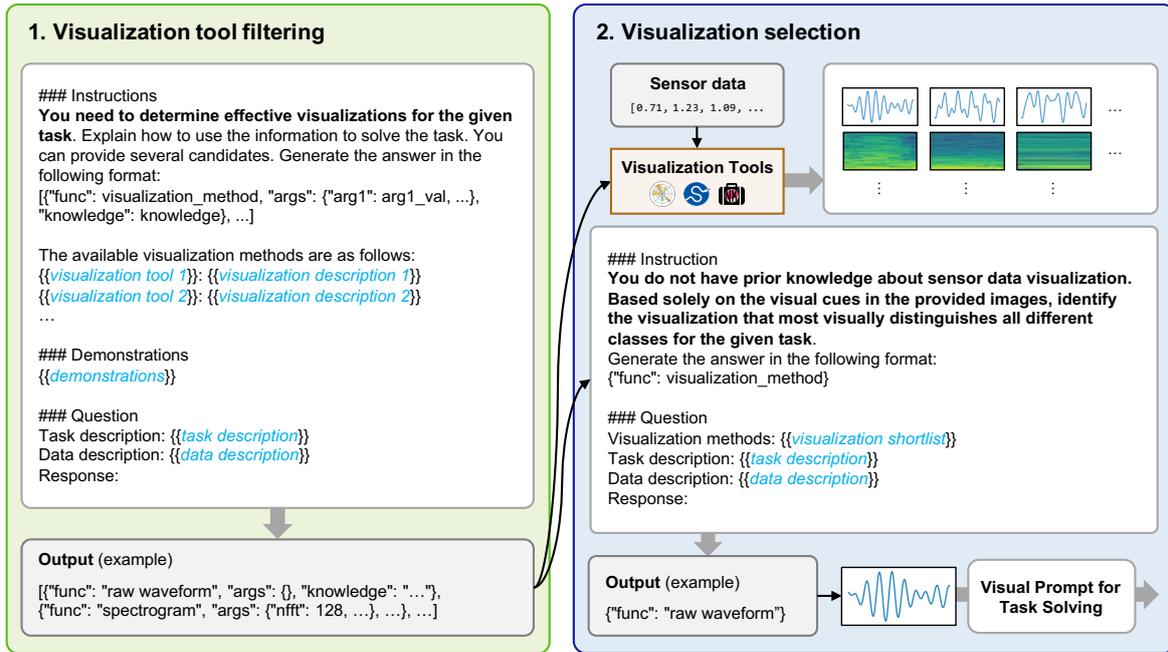


Figure 4: Overview of our visualization generator. First, visualization tool filtering generates a filtered list of visualization tools from public libraries based on the task and data descriptions. Next, visualization selection generates and selects the most effective visualization by asking MLLMs to observe visualized sensor data prepared for the task using all the filtered visualization methods.

315 dure. First, different visualizations are generated
 316 using the filtered tools. With the images, we
 317 instruct the MLLM to select the best visualization
 318 by providing a textual prompt, including the visu-
 319 alization methods, task, and data details. We found
 320 that MLLM often makes incorrect decisions by pri-
 321 oritizing the task description over the visual aids.
 322 To prioritize visual efficacy, we explicitly instruct
 323 the MLLM to avoid relying on prior knowledge
 324 about sensor data and focus on the provided im-
 325 ages. Finally, our automated framework conveys
 326 the selected visualization to the visual prompt for
 327 task solving.

328 5 Experiments

329 We evaluate the applicability of our approach with
 330 MLLMs by conducting experiments on a range of
 331 sensory tasks.

332 5.1 Setups

333 We assume a practical scenario where non-expert
 334 users attempt to solve sensory tasks using MLLMs
 335 (1) without prior knowledge of relevant features
 336 and (2) without external resources to fine-tune the
 337 MLLM. Given the constraints, we leveraged the
 338 few-shot prompting (Brown et al., 2020) approach.
 339 For the main evaluation, we used 1-shot examples

340 where users provide the MLLM with minimal ex-
 341 amples to guide task-solving.

342 **Sensory tasks** We established nine different sen-
 343 sory tasks across four sensor modalities: ac-
 344 celerometer, electrocardiography (ECG) sensor,
 345 electromyography (EMG) sensor, and respiration
 346 sensor. We used three datasets for tasks using ac-
 347 celerometers: HHAR (Stisen et al., 2015) for basic
 348 human activity recognition (running and walking),
 349 UTD-MHAD (Chen et al., 2015) for complex ac-
 350 tivity recognition with fine-grained arm motions,
 351 and a swimming style recognition dataset (Brun-
 352 ner et al., 2019). We use the PTB-XL (Wagner
 353 et al., 2020) dataset for the arrhythmia diagnosis
 354 tasks that use ECG. The dataset includes detection
 355 tasks for four different types of arrhythmia symp-
 356 toms. For EMG data, we used a dataset (Ozdemir
 357 et al., 2022) for hand gesture recognition. Finally,
 358 we used a stress detection task using respiration
 359 sensors provided by the WESAD (Schmidt et al.,
 360 2018) dataset. Details on each task, including the
 361 classes, sampling rates, windowing durations, and
 362 specific configurations, are in Appendix D.

363 **Data processing.** We normalized data using the
 364 mean and standard deviation values calculated for
 365 each user. Test splits were created by randomly
 366 sampling 30 samples per class. For the UTD-
 367 MHAD dataset, we sampled 10 samples per class

368 due to the limited sample availability. Examples of
369 few-shot prompting were randomly sampled, ensur-
370 ing no overlap with the test set. Each task employed
371 the window sizes and sampling rates specified in
372 the original dataset descriptions (see Appendix D).

373 **Baselines.** We set *text-only prompts* for convey-
374 ing sensor data to MLLMs as the main baseline
375 to be compared with our visual prompts. Text-
376 only prompts represented sensor data as numbers
377 within the prompt. We designed text-only prompts
378 by following the latest prompting studies incorpo-
379 rating sensor data into LLMs for healthcare (Liu
380 et al., 2023). Additionally, to establish an upper
381 bound for task-specific performance, we included
382 a fully-supervised baseline using neural networks
383 trained on 75% of the entire data after excluding
384 the test and validation sets. We adopted architec-
385 tures widely accepted for each type of sensor data:
386 1D CNNs for activity recognition with accelerome-
387 ters (Chen et al., 2021) and EMG data (Xiong et al.,
388 2021), as well as for WESAD (Vos et al., 2023),
389 and XResNet-101 for PTB-XL (Strodthoff et al.,
390 2023).

391 **Implementation.** We used GPT-4o from the Ope-
392 nAI API (OpenAI, 2024) as MLLM. The text-only
393 prompts contained the same information as the gener-
394 ated visualization to ensure a fair comparison
395 between text-only and visual prompts. For exam-
396 ple, if the visualization generator outputs a plot
397 with peak notations, the corresponding text-only
398 prompt contains the same features, including the
399 peak values with their indices. When the informa-
400 tion could not fit within the token limit (128K), we
401 used the raw waveform.

402 **Metrics.** We evaluated the experimental results
403 based on accuracy. We also assess the number of
404 tokens used by each prompt method. Tokens are
405 counted using the o200k_base encoding used for
406 GPT-4. To estimate the token cost for images in the
407 same space as text, we follow the token cost com-
408 putation guidelines provided by OpenAI (OpenAI,
409 2024).

410 5.2 Results

411 **Performance.** Table 1 shows the overall perfor-
412 mance of utilizing visual prompts for solving sen-
413 sory tasks. For the same 1-shot prompting, visual
414 prompts consistently showed enhanced accuracies
415 than text-only prompts, achieving an average in-
416 crease of 10%. Notably, the UTD-MHAD task
417 exhibited a significant accuracy gain of up to 33%.
418 See Appendix E for prompt examples with result-

ing visualizations.

419 In addition to achieving higher accuracy, visual
420 prompts are more cost-effective. The number of
421 tokens used for visual prompts in Table 1 shows a
422 substantial reduction, averaging $15.8\times$ fewer than
423 text-only prompts. MLLMs calculate token costs
424 for images within the same token space as text
425 but with distinct counting criteria. In our experi-
426 ments, GPT-4o counts tokens for images based on
427 the number of 512×512 pixel blocks (N) covering
428 the image input, calculated at $85 + 170 \times N$. Our
429 visualized sensor data was represented within a sin-
430 gle 512×512 pixel image, regardless of the sensor
431 data length, significantly reducing costs. Note that
432 the number of tokens from visual prompts is only
433 affected by the number of examples, as all images
434 are the same size. In contrast, text prompts are
435 heavily influenced by high sampling rates and long
436 durations.

437 To further understand the effectiveness of visual
438 prompts with small tokens, we analyzed the infor-
439 mation capacity at the same token cost. Consider-
440 ing a budget of 500 tokens, text-based prompts can
441 include approximately 2,000 ASCII characters. In
442 contrast, visual prompts can input two 512×512
443 px images. In terms of bytes, 2,000 ASCII char-
444 acters amount to 2 KB, whereas two RGB images
445 occupy 1.57 MB, which is $785\times$ larger. Although
446 this calculation does not directly translate to the
447 exact amount of useful information, it suggests that
448 well-designed visual prompts can convey a wider
449 range of information than text prompts within the
450 same cost constraint.

451 **Effect of number of shots.** To investigate the im-
452 pact of varying numbers of shots, we experimented
453 using different numbers of examples (1-shot, 3-
454 shot, and 5-shot) within the prompts. We used the
455 ECG dataset, allowing multiple examples with text-
456 only prompts due to its lower token consumption.

457 Figure 5 depicts the results. Prompting meth-
458 ods are color-coded (blue for visual and green
459 for text-only), and different markers indicate the
460 number of shots. We compared the accuracy and
461 counted the tokens for each setting. We found that
462 visual prompts constantly outperformed text-only
463 prompts with the same number of examples, indi-
464 cating the robustness of our method in different
465 few-shot settings. Additionally, when comparing
466 visual prompts and text-only prompts under the
467 same token budget (5-shot visual prompt versus
468 1-shot text-only prompt), visual prompts often per-
469 formed significantly better (MI and HYP detection).
470

Table 1: Comparison of the text prompts and visual prompts for solving sensory tasks using GPT-4o. The highest accuracy values are highlighted in bold. The visual prompts (multi-shot) utilize the maximum number of examples by matching the token size of the 1-shot text prompts.

Method	Accelerometer			ECG				EMG	Resp	Avg.
	HHAR	UTD-MHAD	Swim	PTB-XL (CD)	PTB-XL (MI)	PTB-XL (HYP)	PTB-XL (STTC)	Gesture	WESAD	
<i>Accuracy</i>										
Task-specific model	0.95	0.95	0.99	0.88	0.86	0.90	0.90	0.64	0.69	0.86
Text-only prompt	0.66	0.10	0.51	0.73	0.62	0.47	0.53	0.27	0.48	0.49
Visual prompt (ours)	0.67	0.43	0.73	0.80	0.68	0.55	0.57	0.30	0.61	0.59
<i>Number of tokens</i>										
Text-only prompt	52910	50439	16586	3204	2766	2757	3596	88655	60253	31244
Visual prompt (ours)	2020	5963	1768	943	943	943	946	3073	1211	1979
	(26.2×↓)	(8.5×↓)	(9.4×↓)	(3.4×↓)	(2.9×↓)	(2.9×↓)	(3.8×↓)	(28.9×↓)	(49.8×↓)	(15.8×↓)

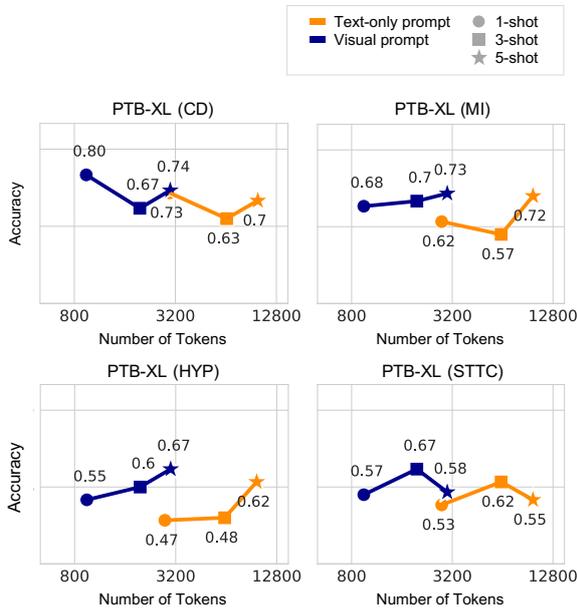


Figure 5: Accuracy of arrhythmia detection tasks using visual and text-only prompts with different shots.

This highlights the advantage of token-efficient visual prompts that can utilize more resources for better performance under the same token constraint.

Unlike our expectations, additional examples did not always result in better performance. This result aligns with existing reports indicating that more examples do not always guarantee better results (Perez et al., 2021; Lu et al., 2021). We further hypothesize that a longer context might hinder the MLLM’s ability to retrieve important information (Liu et al., 2024). Our findings suggest that the impact of shots is data-dependent, and effectively utilizing more examples for consistent improvement remains an open question for further research.

Effect of visualization generator. We conducted

Table 2: Performance of using different visualization methods for visual prompts. A fixed visualization defaulted as raw waveform plots and visualizations selected from the visualization generator are compared. The highest values (or within 0.01) are noted in bold.

Sensor	Dataset	Visualization Method	
		Raw waveform	Visualization Generator
Accel.	HHAR	0.70	0.67
	UTD-MHAD	0.41	0.43
	Swim	0.74	0.73
ECG	PTB-XL (CD)	0.60	0.80
	PTB-XL (MI)	0.58	0.68
	PTB-XL (HYP)	0.53	0.55
	PTB-XL (STTC)	0.53	0.57
EMG	Gesture	0.30	0.30
Resp.	WESAD	0.62	0.61
Average		0.56	0.59

an ablation study to assess the impact of the visualization generator. We compared the visualization generator to a fixed visualization method that defaults to raw waveform plots. Table 2 summarizes the performance comparison and Figure 6 depicts the visualization methods selected by the generator for different tasks.

The selected visualization methods varied depending on the sensors. For tasks involving accelerometer data, the visualization generator mainly selected raw waveforms, occasionally spectrograms. As a result, the performance of the tasks was similar to that of the fixed raw waveform visualizations. On the other hand, when raw waveforms were used to visualize ECG data, results showed performance close to random predictions. Meanwhile, when visualizations are generated through

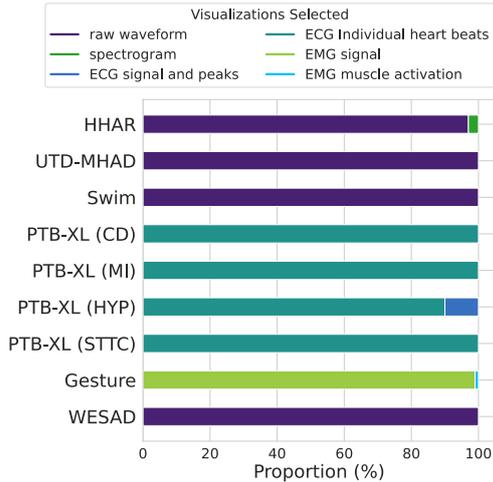


Figure 6: Proportion of selected visualization methods from the visualization generator across different sensory tasks.

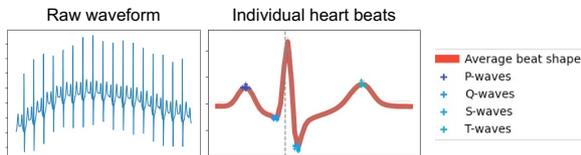


Figure 7: Examples of ECG data visualizations. The visualization generator selected the individual heartbeats plot.

the visualization generator, results showed significantly improved performance, achieving up to a 20% accuracy gain for PTB-XL (CD). This improvement was due to the selection of the specialized visualization method: the individual heartbeat plot with peak notations (Figure 7). These plots provided more insightful visual information for ECG analysis than raw waveforms. For EMG gesture recognition and WESAD tasks, the generator mainly selected muscle activation plots and raw waveforms, respectively, resulting in performance comparable to the fixed approach. Overall, our findings indicate that the visualization generator can lead to significant improvements when the default visualization shows suboptimal performance.

6 Conclusion

We addressed sensory tasks by providing visualized sensor data as images to MLLMs. We designed a visual prompt to instruct MLLMs in using visualized sensor data, provided with textual descriptions of the task and data collection methods. Additionally, we introduced a visualization generator that automatically selects the best visualization method

for each task using visualization tools available in public libraries. We conducted experiments across nine different sensory tasks and four sensor modalities, each with a distinct task. Our results suggest that the visual prompts generated by our visualization generator not only improve accuracy by an average of 10% over text-based prompts but also significantly reduce costs, requiring $15.8\times$ fewer tokens. This indicates that our approach with visual prompts and a visualization generator is a practical solution for general sensory tasks.

Limitations

Our study demonstrates the effectiveness of visual prompts on nine different sensory tasks, primarily focusing on classification. While visual prompts effectively highlight patterns over images, for tasks requiring numerical retrieval or precise computations—where exact values are critical—text prompts can be more effective due to their inclusion of specific numeric data, which are omitted in visual representations. Notably, our approach integrates both images and texts in prompts, allowing the inclusion of numerical values in the text. Determining the optimal distribution of information between images and text to compose a prompt that effectively addresses sensory tasks presents a future direction for this work.

Visualizing sensor data as plots often presents challenges. For instance, brain wave analysis using high-density EEG involves up to 256 channels (Fiedler et al., 2022), complicating their representation in a single visual plot. We denote different channels as distinct notations within a plot, making densely populated plots visually indecipherable. An alternative method of plotting distinct channels across separate subplots was explored but resulted in a significant drop in performance (see Appendix 4). We hypothesize that this limitation arises from the dispersion of information across various areas, highlighting that effective visualization of large-channel datasets remains challenging. This underscores the need for improved visualization techniques in such scenarios.

Our visual prompt design does not incorporate Chain-of-Thought (CoT) prompting (Kojima et al., 2022). Experiments using zero-shot CoT on our datasets revealed inconsistent benefits (see Appendix A), unlike the widely known effect of CoT for enhancing performance. We suspect this may be due to the complexities of reasoning over sen-

577	sory data. Given the observation, further research	DeepMind. 2024. Gemini. https://deepmind.google/technologies/gemini/ .	629
578	is needed to develop methods that effectively inte-		630
579	grate reasoning and interpretation into the decision-		
580	making processes for sensor data analysis.	Ty Ferguson, Timothy Olds, Rachel Curtis, Henry Blake,	631
581	Lastly, the high costs of text-only prompts in	Alyson J Crozier, Kylie Dankiw, Dorothea Dumuid,	632
582	sensory tasks constrained our testing to 30 samples	Daiki Kasai, Edward O’Connor, Rosa Virgara, et al.	633
583	per class. Expanding the scale as resources allow	2022. Effectiveness of wearable activity trackers	634
584	could provide a more robust analysis and poten-	to increase physical activity and improve health: a	635
585	tially validate a broader spectrum of applications.	systematic review of systematic reviews and meta-	636
		analyses. <i>The Lancet Digital Health</i> , 4(8):e615–	637
		e626.	638
586	References		
587	Josh Achiam, Steven Adler, Sandhini Agarwal, Lama	Patrique Fiedler, Carlos Fonseca, Eko Supriyanto, Frank	639
588	Ahmad, Ilge Akkaya, Florencia Leoni Aleman,	Zanow, and Jens Haueisen. 2022. A high-density	640
589	Diogo Almeida, Janko Altenschmidt, Sam Altman,	256-channel cap for dry electroencephalography. <i>Hu-</i>	641
590	Shyamal Anadkat, et al. 2023. Gpt-4 technical report.	<i>man brain mapping</i> , 43(4):1295–1308.	642
591	<i>arXiv preprint arXiv:2303.08774</i> .		
592	Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc,	Marcus Georgi, Christoph Amma, and Tanja Schultz.	643
593	Antoine Miech, Iain Barr, Yana Hasson, Karel	2015. Recognizing hand and finger gestures with	644
594	Lenc, Arthur Mensch, Katherine Millican, Malcolm	imu based motion and emg based muscle activity	645
595	Reynolds, et al. 2022. Flamingo: a visual language	sensing. In <i>International Conference on Bio-inspired</i>	646
596	model for few-shot learning. <i>Advances in Neural</i>	<i>Systems and Signal Processing</i> , volume 2, pages 99–	647
597	<i>Information Processing Systems</i> , 35:23716–23736.	108. Scitepress.	648
598	Kerem Altun and Billur Barshan. 2010. Human activity	Ary L Goldberger, Zachary D Goldberger, and Alexei	649
599	recognition using inertial/magnetic sensor units. In	Shvilkin. 2017. <i>Clinical Electrocardiography: A</i>	650
600	<i>Human Behavior Understanding: First International</i>	<i>Simplified Approach: Clinical Electrocardiography:</i>	651
601	<i>Workshop. Proceedings 1</i> , pages 38–51. Springer.	<i>A Simplified Approach E-Book</i> . Elsevier Health Sci-	652
602	Tom Brown, Benjamin Mann, Nick Ryder, Melanie	ences.	653
603	Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind	Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G	654
604	Neelakantan, Pranav Shyam, Girish Sastry, Amanda	Wilson. 2024. Large language models are zero-shot	655
605	Askeel, et al. 2020. Language models are few-shot	time series forecasters. <i>Advances in Neural Informa-</i>	656
606	learners. <i>Advances in Neural Information Processing</i>	<i>tion Processing Systems</i> , 36.	657
607	<i>Systems</i> , 33:1877–1901.		
608	Gino Brunner, Darya Melnyk, Birkir Sigfússon, and	Tanmay Gupta and Aniruddha Kembhavi. 2023. Visual	658
609	Roger Wattenhofer. 2019. Swimming style recogni-	programming: Compositional visual reasoning with-	659
610	tion and lap counting using a smartwatch and deep	out training. In <i>Conference on Computer Vision and</i>	660
611	learning. In <i>ACM International Symposium on Wear-</i>	<i>Pattern Recognition</i> , pages 14953–14962.	661
612	<i>able Computers</i> , pages 23–31.		
613	Sébastien Bubeck, Varun Chandrasekaran, Ronen El-	Hasan Hayat, Thomas Griffiths, Desmond Brennan,	662
614	dan, Johannes Gehrke, Eric Horvitz, Ece Kamar,	Richard P Lewis, Michael Barclay, Chris Weirman,	663
615	Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lund-	Bruce Philip, and Justin R Searle. 2019. The state-	664
616	berg, et al. 2023. Sparks of artificial general intelli-	of-the-art of sensors and environmental monitoring	665
617	gence: Early experiments with gpt-4. <i>arXiv preprint</i>	technologies in buildings. <i>Sensors</i> , 19(17):3648.	666
618	<i>arXiv:2303.12712</i> .		
619	Chen Chen, Roozbeh Jafari, and Nasser Kehtarnavaz.	John D Hunter. 2007. Matplotlib: A 2d graphics en-	667
620	2015. Utd-mhad: A multimodal dataset for human	vironment. <i>Computing in science & engineering</i> ,	668
621	action recognition utilizing a depth camera and a	9(03):90–95.	669
622	wearable inertial sensor. In <i>International conference</i>	Chihiro Ito, Xin Cao, Masaki Shuzo, and Eisaku Maeda.	670
623	<i>on image processing (ICIP)</i> , pages 168–172. IEEE.	2018. Application of cnn for human activity recog-	671
624	Kaixuan Chen, Dalin Zhang, Lina Yao, Bin Guo, Zhi-	nition with fft spectrogram of acceleration and gyro	672
625	wen Yu, and Yunhao Liu. 2021. Deep learning for	sensors. In <i>ACM international joint conference and</i>	673
626	sensor-based human activity recognition: Overview,	<i>2018 international symposium on pervasive and ubiq-</i>	674
627	challenges, and opportunities. <i>ACM Computing Sur-</i>	<i>uitous computing and wearable computers</i> , pages	675
628	<i>veys (CSUR)</i> , 54(4):1–40.	1503–1510.	676
		Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu,	677
		James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan	678
		Liang, Yuan-Fang Li, Shirui Pan, et al. 2023a. Time-	679
		llm: Time series forecasting by reprogramming large	680
		language models. <i>arXiv preprint arXiv:2310.01728</i> .	681

682	Ming Jin, Qingsong Wen, Yuxuan Liang, Chaoli Zhang,	OpenAI. 2024. Hello gpt-4o. https://openai.com/index/hello-gpt-4o/ .	737
683	Siqiao Xue, Xue Wang, James Zhang, Yi Wang,		738
684	Haifeng Chen, Xiaoli Li, et al. 2023b. Large models		
685	for time series and spatio-temporal data: A survey	Mehmet Akif Ozdemir, Deniz Hande Kisa, Onan Guren,	739
686	and outlook. <i>arXiv preprint arXiv:2310.10196</i> .	and Aydin Akan. 2022. Dataset for multi-channel	740
		surface electromyography (semg) signals of hand	741
687	Yubin Kim, Xuhai Xu, Daniel McDuff, Cynthia	gestures. <i>Data in brief</i> , 41:107921.	742
688	Breazeal, and Hae Won Park. 2024. Health-llm:		
689	Large language models for health prediction via wear-	Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021.	743
690	able sensor digital. In <i>Conference on Health, Infer-</i>	True few-shot learning with language models. <i>Ad-</i>	744
691	<i>ence, and Learning</i> , Proceedings of Machine Learn-	<i>ances in Neural Information Processing Systems</i> ,	745
692	ing Research, pages 1–15. PMLR.	34:11054–11070.	746
693	Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yu-	Susanna Pirttikangas, Kaori Fujinami, and Tatsuo Naka-	747
694	taka Matsuo, and Yusuke Iwasawa. 2022. Large lan-	jima. 2006. Feature selection and activity recognition	748
695	guage models are zero-shot reasoners. <i>Advances in</i>	from wearable sensors. In <i>Ubiquitous Computing</i>	749
696	<i>Neural Information Processing Systems</i> , 35:22199–	<i>Systems: Third International Symposium. Proceed-</i>	750
697	22213.	<i>ings 3</i> , pages 516–527. Springer.	751
698	Li Liu, Yuxin Peng, Shu Wang, Ming Liu, and Zigang	Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta	752
699	Huang. 2016. Complex activity recognition using	Raileanu, Maria Lomeli, Eric Hambro, Luke Zettle-	753
700	time series pattern dictionary learned from ubiquitous	moyer, Nicola Cancedda, and Thomas Scialom. 2024.	754
701	sensors. <i>Information Sciences</i> , 340:41–57.	Toolformer: Language models can teach themselves	755
		to use tools. <i>Advances in Neural Information Pro-</i>	756
702	Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paran-	<i>cessing Systems</i> , 36.	757
703	jape, Michele Bevilacqua, Fabio Petroni, and Percy		
704	Liang. 2024. Lost in the middle: How language mod-	Philip Schmidt, Attila Reiss, Robert Duerichen, Claus	758
705	els use long contexts. <i>Transactions of the Association</i>	Marberger, and Kristof Van Laerhoven. 2018. In-	759
706	<i>for Computational Linguistics</i> , 12:157–173.	roducing wesad, a multimodal dataset for wearable	760
		stress and affect detection. In <i>ACM international con-</i>	761
707	Xin Liu, Daniel McDuff, Geza Kovacs, Isaac Galatzer-	<i>ference on multimodal interaction</i> , pages 400–408.	762
708	Levy, Jacob Sunshine, Jiening Zhan, Ming-Zher Poh,		
709	Shun Liao, Paolo Di Achille, and Shwetak Patel.	Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li,	763
710	2023. Large language models are few-shot health	Weiming Lu, and Yueting Zhuang. 2024. Hugging-	764
711	learners. <i>arXiv preprint arXiv:2305.15525</i> .	gpt: Solving ai tasks with chatgpt and its friends	765
		in hugging face. <i>Advances in Neural Information</i>	766
712	Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-	<i>Processing Systems</i> , 36.	767
713	Wei Chang, Ying Nian Wu, Song-Chun Zhu, and		
714	Jianfeng Gao. 2024. Chameleon: Plug-and-play com-	Allan Stisen, Henrik Blunck, Sourav Bhattacharya,	768
715	positional reasoning with large language models. <i>Ad-</i>	Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind	769
716	<i>advances in Neural Information Processing Systems</i> ,	Dey, Tobias Sonne, and Mads Møller Jensen. 2015.	770
717	36.	Smart devices are different: Assessing and mitigat-	771
		ing mobile sensing heterogeneities for activity recog-	772
718	Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel,	ognition. In <i>ACM conference on embedded networked</i>	773
719	and Pontus Stenetorp. 2021. Fantastically ordered	<i>sensor systems</i> , pages 127–140.	774
720	prompts and where to find them: Overcoming		
721	few-shot prompt order sensitivity. <i>arXiv preprint</i>	Nils Strodthoff, Temesgen Mehari, Claudia Nagel,	775
722	<i>arXiv:2104.08786</i> .	Philip J Aston, Ashish Sundar, Claus Graff, Jør-	776
		gen K Kanters, Wilhelm Haverkamp, Olaf Dössel,	777
723	Dominique Makowski, Tam Pham, Zen J Lau, Jan C	Axel Loewe, et al. 2023. Ptb-xl+, a comprehensive	778
724	Brammer, François Lespinasse, Hung Pham, Christo-	electrocardiographic feature dataset. <i>Scientific data</i> ,	779
725	pher Schölzel, and SH Annabel Chen. 2021. Neu-	10(1):279.	780
726	rokrit2: A python toolbox for neurophysiological sig-		
727	nal processing. <i>Behavior research methods</i> , pages	Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi,	781
728	1–8.	SM Eslami, Oriol Vinyals, and Felix Hill. 2021. Mul-	782
		timodal few-shot learning with frozen language mod-	783
729	Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter,	els. <i>Advances in Neural Information Processing Sys-</i>	784
730	Danny Driess, Montserrat Gonzalez Arenas, Kan-	<i>tems</i> , 34:200–212.	785
731	ishka Rao, Dorsa Sadigh, and Andy Zeng. 2023.		
732	Large language models as general pattern machines.	Yunus Emre Ustev, Ozlem Durmaz Incel, and Cem Er-	786
733	In <i>Conference on Robot Learning</i> , pages 2498–2518.	soy. 2013. User, device and orientation independent	787
734	PMLR.	human activity recognition on mobile phones: Chal-	788
		lenges and a proposal. In <i>ACM conference on Perva-</i>	789
735	OpenAI. 2022. Introducing chatgpt. https://www.	<i>sive and ubiquitous computing adjunct publication</i> ,	790
736	openai.com/blog/chatgpt .	pages 1427–1436.	791

792	Vini Vijayan, James P Connolly, Joan Condell, Nigel McKelvey, and Philip Gardiner. 2021. Review of wearable devices and data collection considerations for connected health. <i>Sensors</i> , 21(16):5589.	Xiyuan Zhang, Ranak Roy Chowdhury, Rajesh K Gupta, and Jingbo Shang. 2024b. Large language models for time series: A survey. <i>arXiv preprint arXiv:2402.01801</i> .	848
793			849
794			850
795			851
796	Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. 2020. Scipy 1.0: fundamental algorithms for scientific computing in python. <i>Nature methods</i> , 17(3):261–272.	Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. 2023. One fits all: Power general time series analysis by pretrained lm. <i>Advances in Neural Information Processing Systems</i> , 36:43322–43355.	852
797			853
798			854
799			855
800			
801			
802	Gideon Vos, Kelly Trinh, Zoltan Sarnyai, and Mostafa Rahimi Azghadi. 2023. Generalizable machine learning for stress monitoring from wearable devices: a systematic literature review. <i>International Journal of Medical Informatics</i> , 173:105026.		
803			
804			
805			
806			
807	Patrick Wagner, Nils Strodthoff, Ralf-Dieter Boussejot, Dieter Kreiseler, Fatima I Lunze, Wojciech Samek, and Tobias Schaeffter. 2020. Ptb-xl, a large publicly available electrocardiography dataset. <i>Scientific data</i> , 7(1):1–15.		
808			
809			
810			
811			
812	Yan Wang, Shuang Cang, and Hongnian Yu. 2019. A survey on wearable sensor modality centred human activity recognition in health care. <i>Expert Systems with Applications</i> , 137:167–190.		
813			
814			
815			
816	Chaoyi Wu, Jiayu Lei, Qiaoyu Zheng, Weike Zhao, Weixiong Lin, Xiaoman Zhang, Xiao Zhou, Ziheng Zhao, Ya Zhang, Yanfeng Wang, et al. 2023. Can gpt-4v (ision) serve medical applications? case studies on gpt-4v for multimodal medical diagnosis. <i>arXiv preprint arXiv:2310.09909</i> .		
817			
818			
819			
820			
821			
822	Dezhen Xiong, Daohui Zhang, Xingang Zhao, and Yiwen Zhao. 2021. Deep learning for emg-based human-machine interaction: A review. <i>Journal of Automatica Sinica</i> , 8(3):512–533.		
823			
824			
825			
826	Huatao Xu, Liying Han, Qirui Yang, Mo Li, and Mani Srivastava. 2024. Penetrative ai: Making llms comprehend the physical world. In <i>International Workshop on Mobile Computing Systems and Applications</i> , pages 1–7.		
827			
828			
829			
830			
831	Hao Xue and Flora D Salim. 2023. Promptcast: A new prompt-based learning paradigm for time series forecasting. <i>Transactions on Knowledge and Data Engineering</i> .		
832			
833			
834			
835	Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. 2023. The dawn of lmms: Preliminary explorations with gpt-4v (ision). <i>arXiv preprint arXiv:2309.17421</i> , 9(1):1.		
836			
837			
838			
839			
840	Han Yu, Peikun Guo, and Akane Sano. 2023. Zero-shot ecg diagnosis with large language models and retrieval-augmented generation. In <i>Machine Learning for Health (ML4H)</i> , pages 650–663. PMLR.		
841			
842			
843			
844	Duzhen Zhang, Yahan Yu, Chenxing Li, Jiahua Dong, Dan Su, Chenhui Chu, and Dong Yu. 2024a. Mm-llms: Recent advances in multimodal large language models. <i>arXiv preprint arXiv:2401.13601</i> .		
845			
846			
847			

A Effect of Zero-shot Chain-of-Thoughts

We experimented with zero-shot Chain-of-Thought (CoT) prompting (Kojima et al., 2022) by adding "let's think step-by-step" to our prompts, testing this on two accelerometers and two ECG datasets. Table 3 shows the findings. While CoT prompting is generally known to enhance LLM response quality, our results showed inconsistent performance by datasets. Notably, CoT consistently dropped performance for text-only prompts. We analyzed the results by observing the CoT responses, illustrated as examples in Figures 8 and Figure 9, showing wrong predictions with CoT from the HHAR dataset. We found that CoT reasoning in text-only prompts primarily focused on simple statistical comparisons, such as whether values were higher or lower. This simplistic approach proved inadequate for analyzing the complexities of sensor data, leading to sub-optimal responses. Likewise, visual prompts indicated reasoning centered around terms like "variations," "periodic," and "stable," but they lacked the necessary depth to effectively assess more intricate features like frequency trends or signal shapes. This superficial reasoning suggests a significant gap in the CoT approach, underscoring the need for more task-specific reasoning prompts for sensory data analysis.

B Use of Subplots for Multi-channel Data

Sensor data often include multiple channels. Our visual prompts differentiated channels using varying colors within a single plot to maintain a shared axis system. To assess the impact of different plotting approaches, we conducted experiments using accelerometer datasets, which have three channels. Specifically, we compared visualizing three distinct plots for each channel against our current approach. Table 4 shows the results. The results indicated that separated plots for each channel reduced performance by 12%. We hypothesize that multiple subplots distribute visual features over different regions, resulting in problems in understanding the relationship between different channels. To this end, we recommend using an aggregated plot when all channels can be represented within a plot. However, for dense datasets, such as 256-channel EEG (Fiedler et al., 2022), a single plot may not suffice, highlighting a limitation in our current visualization approach. Addressing this challenge will be a focus of future research.

Table 3: Performance of text-only and visual prompts, both with and without using CoT. The highest accuracy values are noted in bold.

Prompt	Accel.		ECG		Avg.
	HHAR	Swim	PTB-XL (CD)	PTB-XL (MI)	
Text-only	0.66	0.51	0.73	0.62	0.63
Text-only (CoT)	0.51	0.25	0.63	0.53	0.48
Visual	0.67	0.73	0.80	0.68	0.72
Visual (CoT)	0.63	0.67	0.80	0.73	0.71

Table 4: Performance comparison of visualizing multi-channel sensor data (accelerometer) using a single plot versus multiple subplots. The single plot method combines multiple waveforms in one shared-axis plot, each channel distinguished by color coding.

Plotting approach	HHAR	UTD-MHAD	Swim	Avg.
Single plot	0.67	0.43	0.74	0.61
Multiple subplots	0.53	0.31	0.69	0.51

C Visualization Tools

Our visualization generator employs tools available in public libraries to create visualizations. We have equipped the visualization generator with 16 distinct visualization functions sourced from widely used libraries such as Matplotlib (Hunter, 2007), Scipy (Virtanen et al., 2020), and Neurokit2 (Makowski et al., 2021). The specific visualization tools implemented in our generator and their descriptions are outlined in Table 5. The descriptions presented in the table were directly written inside the prompt for the visualization tool filtering (see Appendix E).

D Details of Sensory Tasks

We conducted experiments across nine sensory tasks across four sensor modalities, each with unique objectives. This section provides the details of these tasks, including task descriptions, classifications, sampling rates, window durations, and data collection protocols. We directly followed the given sampling rate with the original dataset to represent data in text prompts. The descriptions of each dataset are used to formulate the instructions for our visual prompts. The complete prompt examples are in Appendix E.

Human activity recognition: We used the HHAR (Stisen et al., 2015) dataset to classify six basic human activities: sit, stand, walk, bike, up-stairs, and downstairs. Data were collected from

```

To determine the most likely activity from the given sensor data, we need to analyze the patterns and characteristics of the data along the x, y, and z axes. Let's break down the data step-by-step and compare it with the provided examples for each activity.

### Analysis of the Given Data
1. X-axis: The values range from approximately -2.94 to 0.79.
2. Y-axis: The values range from approximately 1.3 to 2.71.
3. Z-axis: The values range from approximately -1.42 to 0.16.

### Comparison with Examples
#### 1. Bike:
- The bike data shows significant variations in all three axes, with values often exceeding 1.0 or -1.0.
- The given data does not show such high variations, especially in the x-axis and z-axis.

#### 2. Sit:
- The sit data shows relatively stable values, especially in the y-axis, which is consistently around -0.8.
- The given data has a y-axis range from 1.3 to 2.71, which is not consistent with sitting.

#### 3. Stand:
- The stand data shows relatively stable values, especially in the y-axis, which is consistently around -0.3.
- The given data has a y-axis range from 1.3 to 2.71, which is not consistent with standing.

#### 4. Walk:
- The walk data shows periodic variations in all three axes, with values often exceeding 1.0 or -1.0.
- The given data does not show such periodic variations, especially in the x-axis and z-axis.

#### 5. Stairsdown:
- The stairsdown data shows significant variations in all three axes, with values often exceeding 1.0 or -1.0.
- The given data does not show such high variations, especially in the x-axis and z-axis.

#### 6. Stairsup:
- The stairsup data shows significant variations in all three axes, with values often exceeding 1.0 or -1.0.
- The given data does not show such high variations, especially in the x-axis and z-axis.

### Conclusion
The given data shows relatively stable values in the x-axis and z-axis, with the y-axis values consistently high (around 1.3 to 2.71). This pattern is most consistent with the activity of sit, where the y-axis values are stable but not as high as in the given data. However, considering the options and the characteristics of the data, the most likely activity is sit.

<answer>sit</answer>

```

Figure 8: An example CoT response from a text-only prompt designed for the HHAR task. The correct prediction is "walk", while the MLLM outputs "sit."

```

Let's analyze the target data step-by-step by comparing it with the provided examples:

1. Sit: The data for sitting shows very little variation in all three axes, with values remaining close to zero. The target data shows significant variation, so it is not sitting.

2. Stand: The data for standing shows some variation, but it is relatively stable compared to other activities. The target data shows more variation than the standing data, so it is unlikely to be standing.

3. Walk: The data for walking shows periodic variations in all three axes. The target data does not show such periodic patterns, so it is unlikely to be walking.

4. Stairsup: The data for walking upstairs shows periodic variations similar to walking but with a different pattern. The target data does not match this pattern.

5. Stairsdown: The data for walking downstairs also shows periodic variations. The target data does not match this pattern.

6. Bike: The data for biking shows variations in all three axes, but it is less periodic and more irregular compared to walking or stairs activities. The target data shows similar irregular variations.

Based on the comparison, the target data most closely resembles the data for biking.

<answer>bike</answer>

```

Figure 9: An example CoT response from a visual prompt designed for the HHAR task. The correct prediction is "stand", while the MLLM outputs "bike."

the built-in accelerometers of smartphones and smartwatches along with the x, y, and z axes. Due to strong domain effects (Ustev et al., 2013), we exclusively used smartwatch data for the experiment. The data, sampled at 100Hz, were segmented into 5-second windows following the established practice for human activity recognition (Altun and Barshan, 2010).

Complex activity recognition: We used the UTD-MHAD (Chen et al., 2015) dataset to classify a wide array of 21 activities: swipe left, swipe right, wave, clap, throw, arms cross, basketball shoot, draw X, draw a circle (clockwise), draw a circle (counter-clockwise), draw a triangle, bowling, boxing, baseball swing, tennis swing, arm curl, tennis serve, push, knock, catch, and pickup and throw. Accelerometers attached to the users’ right wrist were used for data collection. We used data sampled at 50Hz with 3-second windows as described in the dataset documentation.

Swimming style recognition: The swimming dataset (Brunner et al., 2019) involves acceleration data from swimmers performing five different styles: backstroke, breaststroke, butterfly, freestyle, and stationary. This dataset evaluates performance in sports-specific contexts. Data were collected from wrist-worn accelerometers and sampled at 30Hz. We used the 3-second windows recommended with the dataset.

Four arrhythmia detections: The PTB-XL (Wagner et al., 2020) dataset contains ECG recordings from patients with four different arrhythmia types: Conduction Disturbance (CD), Myocardial Infarction (MI), Hypertrophy (HYP), and ST/T Change (STTC). We defined each type as a binary classification task. The dataset comprises 10-second records from clinical 12-lead sensors sampled at 100Hz. We used lead II, the most commonly used lead for arrhythmia detection (Goldberger et al., 2017).

Hand gesture recognition: We included a dataset (Ozdemir et al., 2022) classifying ten different hand gestures using EMG signals: rest, extension, flexion, ulnar deviation, radial deviation, grip, abduction of fingers, adduction of fingers, supination, and pronation. Data were collected from four forearm surface EMG sensors with a 2000Hz sampling rate. We utilized all four channels with a 0.2-second window, following an existing practice known to be effective (Georgi et al., 2015).

Stress Detection: The WESAD (Schmidt et al., 2018) dataset is designed for stress detection (base-

line, stress, amusement) from multiple wearable sensors. We focused exclusively on respiration data measured from the chest for a distinct evaluation setting. The sensor was attached to the users’ chests, with data collected at 700Hz. Following the official guidelines, we employed the three-class classification task (baseline, stress, amusement) using 10-second windows.

E Prompts

We present examples of prompts used in our experiments. Figure 10 and Figure 11 illustrate two text-only prompt examples derived from the HHAR and PTB-XL (CD) datasets; in these examples, sensor data is truncated after a certain point to conserve space, though the format remains consistent with varying values. Figure 12 and Figure 13 displays the visual prompts created for the same datasets, HHAR and PTB-XL (CD). Figure 14 details the prompt for our visualization tool filtering specific to the PTB-XL (CD) task, with demonstrations omitted and presented separately in Figure 15. Lastly, Figure 16 showcases the visualization selection prompt for the PTB-XL (CD) dataset.

Instruction

You are an expert in sensor data analysis. Given the sensor data, determine the correct answer from the options listed in the question. Provide the answer with the format of <answer>ANSWER</answer>, where ANSWER corresponds to one of the options listed in the question. If the answer is not in the options, choose the most possible option.

The sensor data is collected from an accelerometer measuring acceleration along the x, y, and z axes. The data is normalized with the statistics of the user's data. The data is collected over 5 seconds. The data is measured from a smartwatch which was attached to the wrist of a user. Please refer to the provided examples and use them to answer the following question for the target data.

Examples

Example of stand:

Given sensor data (list of ['X-axis', 'Y-axis', 'Z-axis']): [[-0.75, 0.47, -1.27], [-0.75, 0.51, -1.28], [-0.75, 0.49, -1.26], ...

Example of sit:

Given sensor data (list of ['X-axis', 'Y-axis', 'Z-axis']): [[0.78, -1.36, 1.11], [0.78, -1.37, 1.11], [0.78, -1.37, 1.11], ...

Example of walk:

Given sensor data (list of ['X-axis', 'Y-axis', 'Z-axis']): [[-0.45, 0.44, -1.01], [-0.5, 0.6, -0.99], [-0.39, 0.54, -0.98], ...

Example of stairsup:

Given sensor data (list of ['X-axis', 'Y-axis', 'Z-axis']): [[2.87, 0.65, -3.29], [2.75, 0.77, -3.43], [2.59, 0.86, -3.39], ...

Example of stairsdown:

Given sensor data (list of ['X-axis', 'Y-axis', 'Z-axis']): [[-0.96, -0.32, 0.38], [-1.25, 0.02, 0.74], [-1.26, 0.09, 0.58], ...

Example of bike:

Given sensor data (list of ['X-axis', 'Y-axis', 'Z-axis']): [[-0.44, 0.67, 0.59], [-0.64, 1.03, 0.97], [-0.84, 1.68, 0.05], ...

Question

Given sensor data (list of ['X-axis', 'Y-axis', 'Z-axis']): [[-0.38, 0.43, -0.88], [-0.16, 0.63, -0.91], [-0.19, 0.56, -0.91], ...

Question: When the sensor data is used for a task for classifying 6 human activities, bike, sit, stand, walk, stairsdown, stairsup, using three-axis accelerometer data measured from a wrist-worn smartwatch, what is the most likely answer among ['bike', 'sit', 'stand', 'walk', 'stairsdown', 'stairsup']?

Answer:

Figure 10: An example of a text-only prompt for solving the HHAR task. The sensor data represented in the text are truncated beyond a certain point.

Instruction

You are an expert in sensor data analysis. Given the sensor data, determine the correct answer from the options listed in the question. Provide the answer with the format of <answer>ANSWER</answer>, where ANSWER corresponds to one of the options listed in the question. If the answer is not in the options, choose the most possible option.

The ECG data is collected from a lead II ECG sensor. The ECG data is recorded over 10 seconds. The data is normalized with the statistics of the user's data. Please refer to the provided examples and use them to answer the following question for the target data.

Examples

Example of normal:

Average heartbeat in the ECG signal (list of ['lead II']): [-0.34, -0.34, -0.35, -0.35, -0.36, ...
ECG_P_Peaks in the ECG signal (list of (index, value)): [(21, 0.08), (129, 0.19), (239, 0.22), ...
ECG_Q_Peaks in the ECG signal (list of (index, value)): [(35, -0.61), (137, -0.46), (246, -0.48), ...
ECG_S_Peaks in the ECG signal (list of (index, value)): [(42, -1.4), (149, -1.35), (260, -1.33), ...
ECG_T_Peaks in the ECG signal (list of (index, value)): [(63, 2.18), (171, 2.11), (282, 2.31), ...

Example of conduction disturbance:

Average heartbeat in the ECG signal (list of ['lead II']): [-0.15, -0.24, -0.29, -0.31, -0.28, ...
ECG_P_Peaks in the ECG signal (list of (index, value)): [(4, 0.14), (57, 0.3), (103, 0.22), ...
ECG_Q_Peaks in the ECG signal (list of (index, value)): [(14, -0.05), (65, -0.05), (109, -0.07), ...
ECG_S_Peaks in the ECG signal (list of (index, value)): [(22, -2.28), (73, -2.1), (124, -2.35), ...
ECG_T_Peaks in the ECG signal (list of (index, value)): [(82, 0.03), (142, 0.64), (245, 0.25), ...

Question

Average heartbeat in the ECG signal (list of ['lead II']): [-0.39, -0.39, -0.39, -0.39, -0.4, ...
ECG_P_Peaks in the ECG signal (list of (index, value)): [(15, 0.14), (94, -0.26), (173, -0.23), ...
ECG_Q_Peaks in the ECG signal (list of (index, value)): [(23, -0.27), (102, -0.81), (182, -0.55), ...
ECG_S_Peaks in the ECG signal (list of (index, value)): [(34, 0.15), (116, -0.51), (192, -0.45), ...
ECG_T_Peaks in the ECG signal (list of (index, value)): [(50, 1.39), (130, 1.1), (209, 1.31), ...

Question: When the sensor data is used for a task for classifying ECG data into 2 categories: conduction disturbance, normal, what is the most likely answer among ['conduction disturbance', 'normal']?

Answer:

Figure 11: An example of a text-only prompt for solving the PTB-XL (CD) task. The sensor data represented in the text are truncated beyond a certain point.

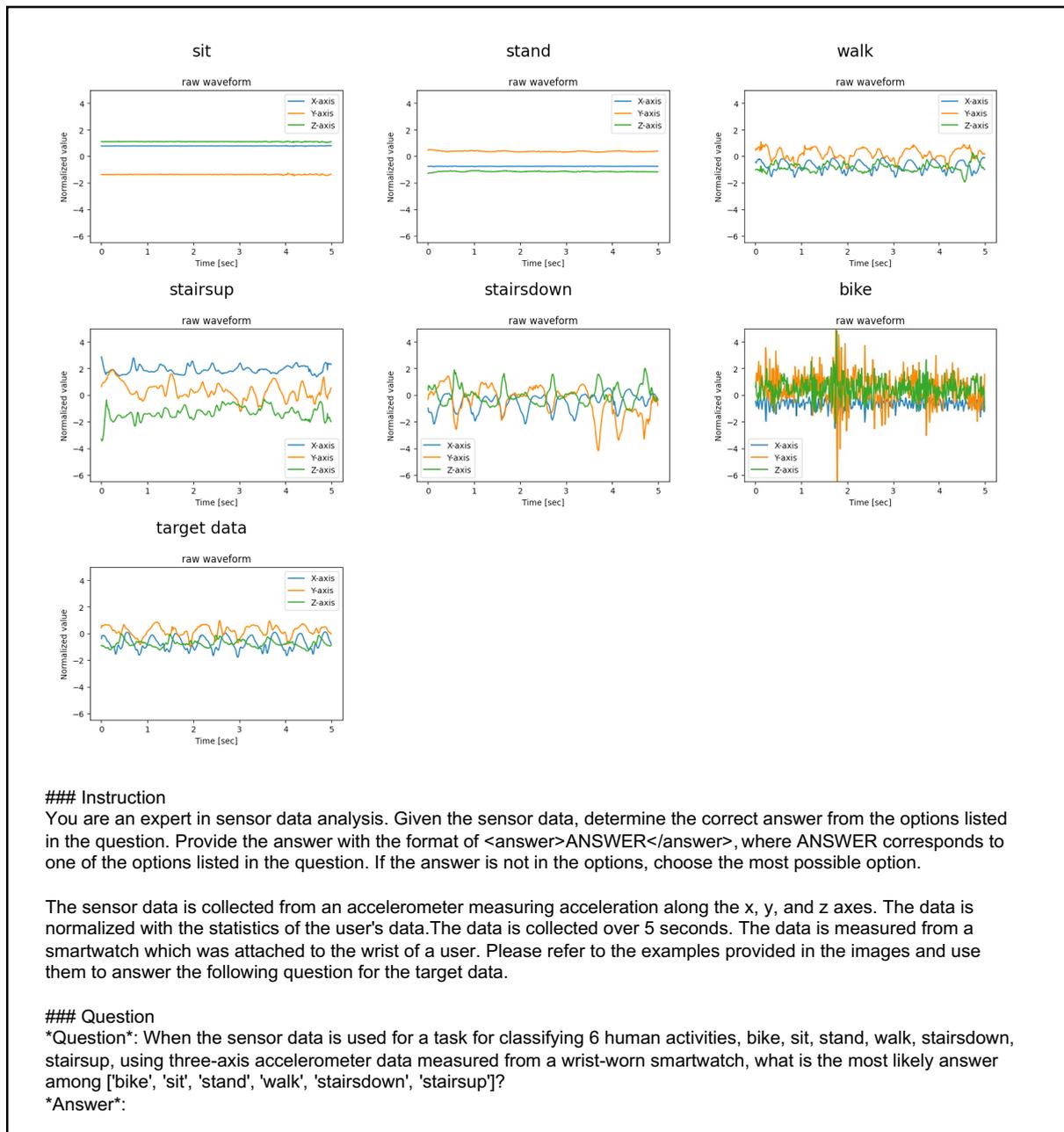


Figure 12: An example of a visual prompt for solving the HHAR task.

Demonstrations

Data description: The sensor data is collected from an accelerometer measuring acceleration along the x, y, and z axes. The data is normalized with the statistics of the user's data. The data is measured from an accelerometer attached to the ankles of a user.

Task description: A task for classifying running and walking activities using accelerometer data measured from an ankle-worn device.

Response: {"func": "raw waveform", "args": {}, "knowledge": "Use this to visualize the amplitude of the accelerometer signal over time. For classifying running and walking, observe the patterns in the waveform: running typically shows higher amplitude and more frequent peaks due to the higher impact and faster motion, while walking shows lower amplitude and less frequent peaks."}

Data description: The sensor data is collected from an accelerometer measuring acceleration along the x, y, and z axes. The data is normalized with the statistics of the vehicle's data. The data is measured from an accelerometer attached to a vehicle.

Task description: A task for classifying road types, such as asphalt, dirt, and cobblestone, using accelerometer data measured from a vehicle.

Response: {"func": "spectrogram", "args": {"nfft": 128, "nperseg": 128, "noverlap": 120, "mode": "magnitude"}, "knowledge": "Use this to analyze the frequency components of the accelerometer signal over time. The colors in the spectrogram represent the magnitude of the frequencies: brighter colors indicate higher magnitudes. For road type classification, asphalt typically shows lower frequency components with smoother patterns, dirt shows higher frequency components with irregular patterns, and cobblestone shows high-frequency components with periodic patterns due to the regular bumps."}

Data description: The sensor data is collected from an ECG measuring the electrical activity of the heart. The data is normalized with the statistics of the user's data. The data is measured using electrodes attached to the chest of a user.

Task description: A task for detecting sleep apnea using ECG data measured from chest electrodes.

Response: {"func": "ECG individual heart beats", "args": {}, "knowledge": "Use this to aggregate and visualize individual heartbeats within an ECG recording. In normal beats, the P-wave precedes the QRS complex, and the T-wave follows it. In sleep apnea, irregularities in the intervals between the P, Q, R, S, and T peaks can be observed. For instance, the absence of regular QRS complexes or prolonged intervals can indicate episodes of apnea. The plot helps identify these patterns by showing the average shape of the heartbeats and marking the specific peaks."}

Data description: The sensor data is collected from an EMG sensor measuring muscle electrical activity. The data is normalized with the statistics of the user's data. The data is measured using electrodes attached to the forearm of a user.

Task description: A task for recognizing finger gestures, such as numbers, using EMG data measured from forearm electrodes.

Response: {"func": "EMG signal", "args": {}, "knowledge": "Use this to visualize the raw EMG signal over time. For recognizing finger gestures, observe the patterns and amplitude of muscle activity. Different numbers (gestures) will produce distinct patterns in the EMG signal. For example, bending more fingers usually results in higher amplitude signals due to increased muscle activation."}

Data description: The sensor data is collected from an ECG measuring the electrical activity of the heart. The data is normalized with the statistics of the user's data. The data is measured using electrodes attached to the chest of a user.

Task description: A task for detecting whether the user is running or not using ECG data measured from chest electrodes.

Response: {"func": "ECG heart rate", "args": {}, "knowledge": "Use this to monitor heart rate over time and analyze activity levels. A significant increase in heart rate can indicate that the user is running. The plot should show a higher average heart rate during running periods compared to resting or walking periods. Sudden spikes and consistent high heart rates are typical indicators of running."}

Figure 15: Demonstrations provided inside the visualization tool filtering prompt to enhance the response quality.

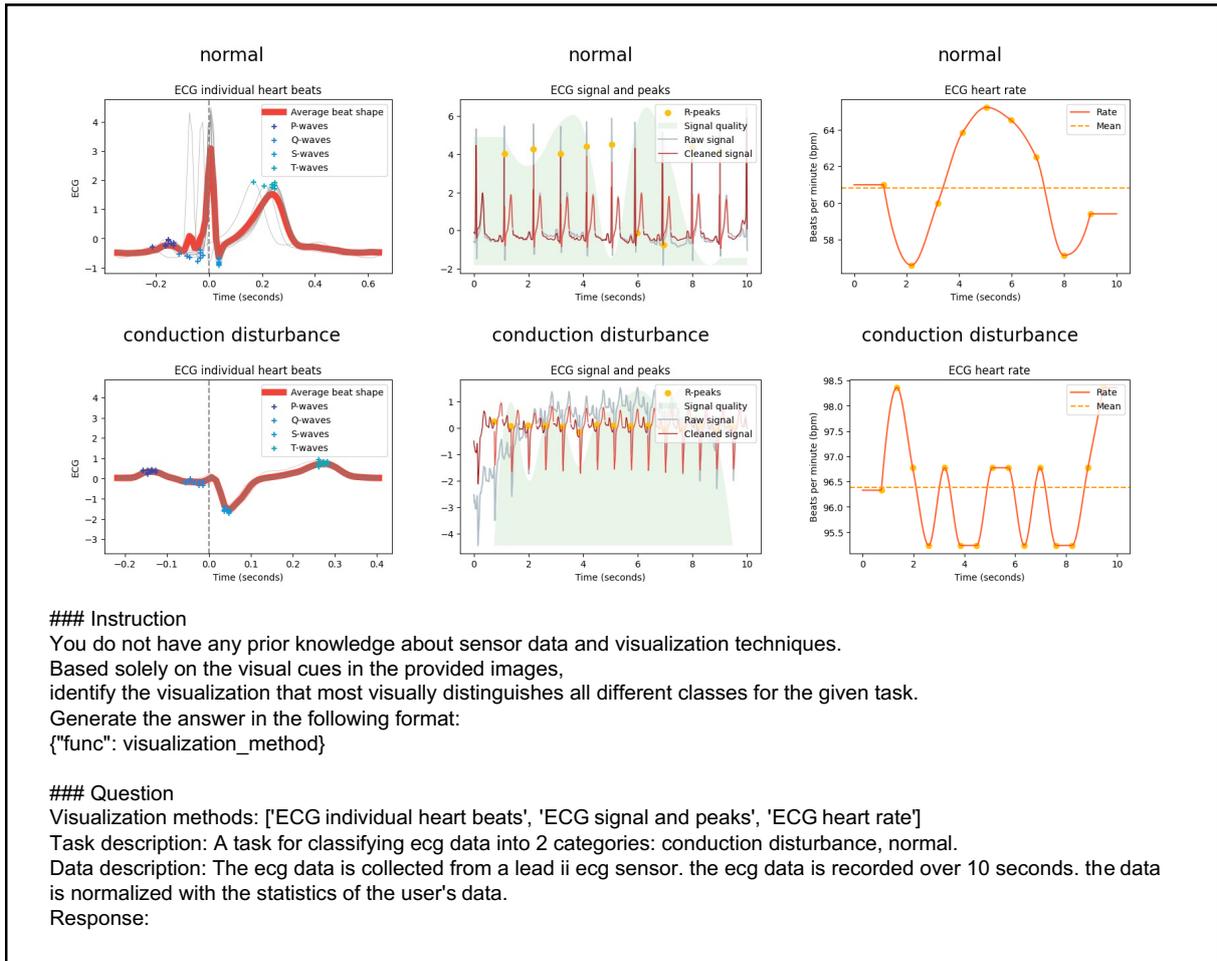


Figure 16: An example prompt from our visualization generator for visualization selection in the PTB-XL (CD) task.

Table 5: Descriptions of the visualization tools provided to our visualization generator.

Visualization tool	Description
raw waveform	This generates a raw signal of sensor data, displaying the amplitude of the signal over time. This is usually used to visualize the raw data and identify patterns in the signal.
spectrogram	This generates a spectrogram of sensor data, showing the density of frequencies over time. This is usually used to visualize the frequency components for high-frequency data, which has features over components but is hard to figure out in the raw plot. It takes the length of the FFT used (nfft), the length of each segment (nperseg), and the number of points to overlap between segments (noverlap) as parameters. Different modes (mode) can be defined to specify the type of return values: ["psd" for power spectral density, "complex" for complex-valued STFT results, "magnitude" for absolute magnitude, "angle" for complex angle, and "phase" for unwrapped phase angle]. (Arguments: nfft, nperseg, noverlap, mode)
signal power spectrum density	This generates a power spectrum density plot, which shows the power of each frequency component of the signal on the x-axis. This is usually used to analyze the signal's power distribution of different frequency components.
EDA signal	This generates a plot showing both raw and cleaned Electrodermal Activity (EDA) signals over time. This is usually used to analyze the EDA signals for patterns related to stress, arousal, or other psychological states.
EDA skin conductance response (SCR)	This generates a plot of skin conductance response (SCR) for EDA data, highlighting the phasic component, onsets, peaks, and half-recovery times. This is usually used to study the transient responses in EDA data related to specific stimuli or events.
EDA skin conductance level (SCL)	This generates a plot of skin conductance level (SCL) for EDA data over time. This is usually used to analyze the tonic component of EDA data, reflecting the overall level of arousal or stress over a period.
ECG signal and peaks	This generates a plot for Electrocardiogram (ECG) data, showing the raw signal, cleaned signal, and R peaks marked as dots to indicate heartbeats. This is usually used to analyze the heartbeats and detect anomalies in the ECG signal.
ECG heart rate	This generates a heart rate plot for ECG data, displaying the heart rate over time and its mean value. This is usually used to monitor and analyze heart rate variability and trends over time.
ECG individual heartbeats	This generates a plot of individual heartbeats and the average heart rate for ECG data. It aggregates heartbeats within an ECG recording and shows the average beat shape, marking P-waves, Q-waves, S-waves, and T-waves. This is usually used to study the morphology of individual heartbeats and identify irregularities.
PPG signal and peaks	This generates a plot for Photoplethysmogram (PPG) data, showing the raw signal, cleaned signal, and systolic peaks marked as dots. This is usually used to analyze the blood volume pulse and detect anomalies in the PPG signal.
PPG heart rate	This generates a heart rate plot for PPG data, displaying the heart rate over time and its mean value. This is usually used for PPG data to monitor and analyze heart rate variability and trends over time.
PPG individual heartbeats	This generates a plot of individual heartbeats and the average heart rate for PPG data, aggregating individual heartbeats within a PPG recording and showing the average beat shape. This is usually used to study the morphology of individual heartbeats based on PPG data.
EMG signal	This generates a plot showing both raw and cleaned Electromyogram (EMG) signals over time. This is usually used to analyze muscle activity and identify patterns in muscle contractions.
EMG muscle activation	This generates a muscle activation plot for EMG data, displaying the amplitudes of muscle activity and highlighting activated parts with lines. This is usually used to study muscle activation levels and identify specific periods of muscle activity.
EOG signal	This generates a plot showing both raw and cleaned Electrooculogram (EOG) signals over time, with blinks marked as dots. This is usually used to analyze eye movement patterns and detect blinks.
EOG blink rate	This generates a blink rate plot for EOG data, displaying the blink rate over time and its mean value. This is usually used to monitor and analyze the blink rate and detect irregularities.
EOG individual blinks	This generates a plot of individual blinks for EOG data, aggregating individual blinks within an EOG recording and showing the median blink shape. This is usually used to study the morphology of individual blinks and identify patterns in blink dynamics.