AutoLife: Automatic Life Journaling with Smartphones and LLMs

HUATAO XU, The Hong Kong University of Science and Technology, Hong Kong SAR
ZILIN ZENG, The Hong Kong University of Science and Technology, Hong Kong SAR
PANRONG TONG, Alibaba Group, China
MO LI*, The Hong Kong University of Science and Technology, Hong Kong SAR
MANI SRIVASTAVA, University of California Los Angeles, USA

This paper introduces a novel mobile sensing application - *life journaling* - designed to generate semantic descriptions of users' daily lives. We present AutoLife, an automatic life journaling system based on commercial smartphones. AutoLife only inputs low-cost sensor data (without photos or audio) from smartphones and can automatically generate comprehensive life journals for users. To achieve this, we first derive time, motion, and location contexts from multimodal sensor data, and harness the zero-shot capabilities of Large Language Models (LLMs), enriched with commonsense knowledge about human lives, to interpret diverse contexts and generate life journals. To manage the task complexity and long sensing duration, a multilayer framework is proposed, which decomposes tasks and seamlessly integrates LLMs with other techniques for life journaling. This study establishes a real-life dataset as a benchmark and extensive experiment results demonstrate that AutoLife produces accurate and reliable life journals. We also demonstrate the real-world application of AutoLife by integrating it into a digital journaling app that automatically generates journals for users.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing systems and tools**; • **Computing methodologies** → *Machine learning approaches.*

Additional Key Words and Phrases: Mobile Sensing, Life Journaling, Large Language Model

ACM Reference Format:
Huatao Xu, Zilin Zeng, Panrong Tong, Mo Li, and Mani Srivastava. 2025. AutoLife: Automatic Life Journaling with Smartphones and LLMs. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 9, 4, Article 226 (December 2025), 29 pages. https://doi.org/10.1145/3770683

1 INTRODUCTION

The widespread adoption of mobile devices like smartphones has significantly transformed many aspects of daily life. Beyond traditional mobile applications, this paper introduces a novel mobile sensing application named "**Life Journaling**" – *an approach to automatically generate detailed semantic and factual descriptions of a person's daily life*. Fig. 1 presents an example of a journal generated from such an envisioned life journaling application, which offers natural and semantic descriptions of the person's life context, including key activities, behaviors, and circumstances in a comprehensive way. We believe life journaling is a very useful application and can support

*Mo Li is the Corresponding author.

Authors' Contact Information: Huatao Xu, The Hong Kong University of Science and Technology, Hong Kong SAR, huatao@ust.hk; Zilin Zeng, The Hong Kong University of Science and Technology, Hong Kong SAR, zzengar@connect.ust.hk; Panrong Tong, Alibaba Group, China, panrong.tpr@alibaba-inc.com; Mo Li, The Hong Kong University of Science and Technology, Hong Kong SAR, lim@ust.hk; Mani Srivastava, University of California Los Angeles, USA, mbs@ucla.edu.

ACM 2474-9567/2025/12-ART226
https://doi.org/10.1145/3770683

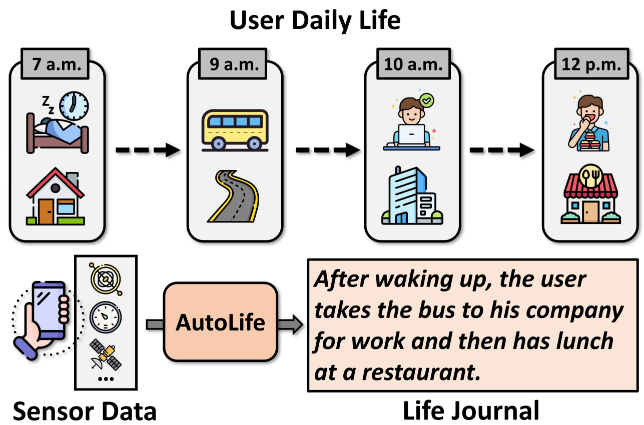


Fig. 1. Life journaling application.

numerous downstream use cases, including personalized recommendations based on user behaviors, automatic annotation or organization of personal photos or video clips based on daily lives, optimizing daily routines for health, and many more.

Unfortunately, to the best of our knowledge, there is no existing solution for such a valuable application at present. Existing lifelogging systems [14, 29, 37] focus on recording daily life as raw digital data such as videos or sensor readings rather than understanding high-level life semantics. Prior human activity recognition (HAR) studies [30, 68, 73, 74, 81, 85] attempt to identify user activities by predicting motion labels like "walking" or "jogging", which are far less informative compared to generating rich life contexts as targeted by life journaling. While there are several commercial digital journaling apps, such as Day One [38] and Journal [31], they are not designed to automatically generate journals and rely heavily on human inputs. So, there is a significant gap in building a viable life journaling system at present.

To fill the gap, this paper presents **AutoLife**, an automatic life journaling system that generates journals of users' daily lives based on smartphone sensor data. A key feature is that AutoLife requires no user input — all a user needs to do is to carry their own smartphone while going about their activities. As shown in Fig. 1, AutoLife processes various sensor readings and other data sources (without photos or audio) accessible from the smartphone, outputting detailed journals of the user's daily life. An essential challenge faced in developing such a system is ***how to fuse those multimodal sensor inputs and generate accurate yet open-vocabulary semantic and factual descriptions?***

To the best of our knowledge, there is no existing dataset for this specific task, making conventional deep-learning solutions inapplicable. Extensive human life knowledge may be required to interpret diverse contexts, e.g., motion and time, and accordingly infer complex human behaviors. This paper builds on our key observation that such context interpretation and inference tasks align well with the strengths of Large Language Models (LLMs), which are trained on large-scale text corpora and possess extensive commonsense knowledge of human behaviors. However, directly using LLMs to analyze sensor data for life journaling can result in hallucinations or low-quality journals due to the high complexity of the task. To address this, our key approach is to extract rich and accurate contexts from sensors, fuse them as flexible texts, and leverage LLMs to synthesize comprehensive life journals from these contextual inputs. Two technical challenges are addressed in the design of AutoLife.

First, we must address a critical question, namely, *what information is desired to derive accurate life journals and how such information can be extracted from various data sources*? While numerous HAR studies [23, 30, 32, 73, 74] have been conducted, we notice that they typically produce only basic motion labels, such as "stationary" or "walking", due to limitations in sensor datasets and the constraints of motion sensors. Such motion contexts can provide some insights into user behaviors but are insufficient for generating a comprehensive journal. In AutoLife, we incorporate two additional contexts - time and location. Both are instrumental in understanding user behaviors, as illustrated in Fig. 1. For instance, if a user remains stationary at a restaurant during mid-noon, it can be reasonably inferred that they are likely having lunch. To detect location context, we exploit GPS locations with geographic information systems (GIS), e.g., the Google Maps Platform [27]. While existing APIs do not reveal comprehensive location contexts, in AutoLife we propose to utilize large vision-language models (VLMs) like GPT-4o [52] to generate location context by interpreting map segments queried from GIS. We also incorporate WiFi SSID information and leverage lighter-weight LLMs like GPT-3.5 to further infer the user's surrounding environment (often when indoors).

Second, to further improve the quality of journals, we build special enhancements around the LLMs, including providing journal examples in the prompts and utilizing two LLM-based modules to pre-process the contexts and post-process the generated journals. Specifically, we address a key challenge of *how to assist LLMs in handling lengthy sensor data collected over long daily life periods?* Different from existing HAR applications interested in labeling short periods of activities [8] like a few seconds, life journaling typically spans a much longer duration over hours, which adds not only complexity to the task, but also difficulties to LLMs in handling the lengthy sensor inputs. To address this challenge, we design a multi-layer framework that breaks life journaling into smaller and manageable subtasks. AutoLife first segments the sensor data into small windows and extracts both motion and location contexts from these segments with the combined use of conventional signal processing and LLM/VLMs. In the middle layer, AutoLife represents the contexts as text, which are then fused and refined before being sent for comprehension by the LLMs. In the last layer, the refined contexts with reduced lengths are consolidated, encapsulating extended-duration context, and finally fed to LLMs to generate the final journals. A duty-cycled data collection approach is applied to further reduce system overhead.

The proposed AutoLife framework is prototyped and evaluated with a self-collected human life dataset that contains diverse behaviors like hiking, cycling, shopping, and working of 10 volunteers with 11 devices. An Android data collection app is developed to continuously collect sensor data from smartphones while users go about their daily activities. For each experiment, the volunteer manually creates reference journals, consisting of text descriptions of the volunteer's behaviors. To evaluate the qualities of journals generated by AutoLife, we compare the similarities between them with the reference journals using metrics such as BERTScore [82]. Our extensive experiments show that certain LLMs, such as Claude 3 combined with our framework, can achieve an average BERTScore F1 exceeding 0.7.

Furthermore, we demonstrate the real-world applicability of AutoLife by developing a journaling app, AutoJournal, which integrates AutoLife with photos and other contextual information to automatically generate daily journals for users. We also highlight key features of the app and present a user study to evaluate its performance and overall user experience. In summary, this paper makes the following contributions:

(1) The paper, for the first time, showcases a novel mobile sensing application that can automatically generate life journals with commercial smartphones, following the concept of Penetrative AI [72].
(2) We present the first life journaling system, AutoLife, which creatively incorporates both LLM/VLMs and conventional signal processing to fuse various sensor data and synthesize long-duration life journals.
(3) The framework is prototyped and comprehensively evaluated. A journaling app is developed to showcase the practical value of AutoLife. Our dataset and benchmark is made publicly available[1] and may serve as a benchmark for future research on this topic.

[1] https://github.com/WANDS-HKUST/AutoLife

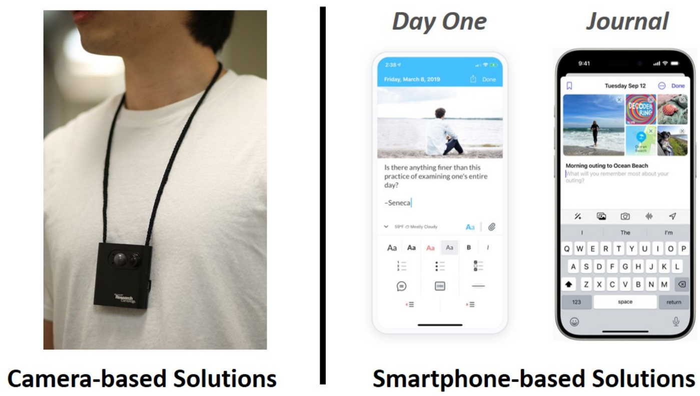


Fig. 2. Existing lifelogging solutions. Left shows a user wears SenseCam [22] while right shows two digital diary applications, i.e., Day One [38] and Journal [31].

The rest of this paper is organized as follows. Section 2 presents the related works. Sections 3-6 introduce the design of AutoLife. Section 7 provides implementation and evaluation results. Section 8 presents the design of the journaling app based on AutoLife. Section 9 discusses, and Section 10 concludes this paper.

2 RELATED WORKS

2.1 Life Logging

Lifelogging [14, 20, 29, 37] is a technique that digitizes human daily life, which can support many applications, including health monitoring and memory enhancement. With the rapid proliferation of mobile devices, many mobile devices or applications have been developed for lifelogging. For example, Microsoft's SenseCam [44] is a pioneering wearable camera designed to capture continuous photographic or video records of a person's day. However, most lifelogging works aim at 'logging' the user's daily life instead of generating high semantic journals. Additionally, many solutions require wearable cameras [10, 28, 81] or smart glasses [34], which are not ubiquitous and introduce extra costs.

Smartphones are widely available and there are numerous digital journaling applications on the market, as illustrated in Fig. 2. However, all these apps require extensive manual input from users. A recent work, MindScape [46, 47] proposes to generate personalized prompts with LLMs, such as "Your running routine has really taken off! How's that influencing your day?" and records the user's responses for journaling, which still requires user input. Unlike existing solutions, our approach generates life journals for users by leveraging sensor data collected from ubiquitous devices like smartphones, eliminating the need for manual input.

2.2 Activity Recognition

Beyond lifelogging, Human activity recognition (HAR) is a critical research topic that aims at recognizing users' daily activities like 'answering the phone' or 'walking'. There are extensive HAR studies and wearable-based solutions [23, 30, 32, 39, 57, 59, 73, 74, 85] can be implemented on off-the-shelf smart devices and are more ubiquitous compared with vision-based [56, 68, 81] or wireless-based [35, 77, 83] solutions.

Despite significant progress in the field, several limitations persist: (1) Most existing methods [23, 30, 32, 33, 73, 74, 85] rely solely on motion sensors like inertial measurement units (IMUs), which are insufficient for distinguishing complex activities. For example, IMU data may only indicate that a user remains stationary for an extended period, without providing enough context to determine whether they are having a meal or attending a class. (2) No existing HAR models can generally and accurately recognize a wide range of motion types, primarily due to the lack of large-scale and comprehensive datasets. More importantly, motion labels obtained from existing HAR methods, such as 'walking' or 'cycling', do not provide the comprehensive information that life journals offer. In summary, current HAR approaches fall short of achieving the goals of life journaling.

2.3 Context Awareness

Location awareness refers to the ability of devices to detect their geographical positions while context awareness [40, 78] extends beyond simple geographical location, allowing devices or systems to interpret various aspects of their environment. Understanding location context is crucial for sensing user behaviors; for example, if a user remains stationary in a restaurant for an extended period, they are likely having a meal. In this paper, we explore a specific aspect of context awareness – "detecting the location context of devices" such as identifying whether a device is at a restaurant or a park. One approach might involve leveraging computer vision models to analyze photos and derive location contexts or scenes [13, 63, 69, 84]. However, it is impractical to expect users to continuously capture photos to generate journals. Instead, this paper introduces a novel method to derive location contexts using low-cost and easily accessible sensor data from smartphones.

2.4 Penetrative AI

Large Language Models (LLMs) have achieved remarkable advancements across a wide range of tasks [12, 45, 50, 60, 65, 79]. These out-of-the-box capabilities demonstrate that LLMs contain vast amounts of world knowledge, acquired through extensive training on large-scale text datasets. Some works [11, 19, 36, 52, 54, 70, 76] extend LLMs into multimodal models, such as vision language models (VLMs) [36], to tackle various image-related tasks. Additionally, several studies introduce innovative LLM applications [7, 41, 72, 75] with sensor data in diverse fields, such as Liu et al.'s work [41], which analyzes medical data for health-related tasks. Notably, researchers have proposed the concept of Penetrative AI [72], exploring the integration of LLMs with the physical world through IoT sensors. With embedded extensive commonsense knowledge, LLMs/VLMs can perform physical tasks by analyzing IoT signals, such as detecting heartbeats using digitized or figure-based ECG data [72]. Inspired by the idea of Penetrative AI, we propose a new application of LLMs/VLMs for deriving life journals from sensor data on smartphones.

3 AUTOLIFE

3.1 Problem Definition

In this paper, we introduce a new application called **life journaling**, which generates journals for users' daily lives through mobile devices. We assume that our system functions as a mobile application on these devices, with regular access to sensor data. The system takes low-cost and long-term sensor data as input, such as accelerometer readings or GPS locations. The output is a series of sentences that accurately describe the user's daily activities, e.g., visiting a museum or resting at home.

3.2 Overview

Fig. 3 presents the overview of AutoLife. Instead of directly feeding long-duration sensor data to LLMs for life journaling, that may cause hallucinations and low-quality journals, AutoLife optimizes the use of LLMs with various sensor data by a multi-layer framework that decomposes the life journaling task process into manageable subtasks, each addressed by specialized modules. First, AutoLife periodically accesses sensor data from smartphones in short periods. The *motion context detection* and *location context detection*, are designed to derive the user's contexts from multiple sensor resources. Particularly, *location context detection* presents a novel approach to obtain accurate and general location contexts using LLMs or VLMs. Next, AutoLife represents these

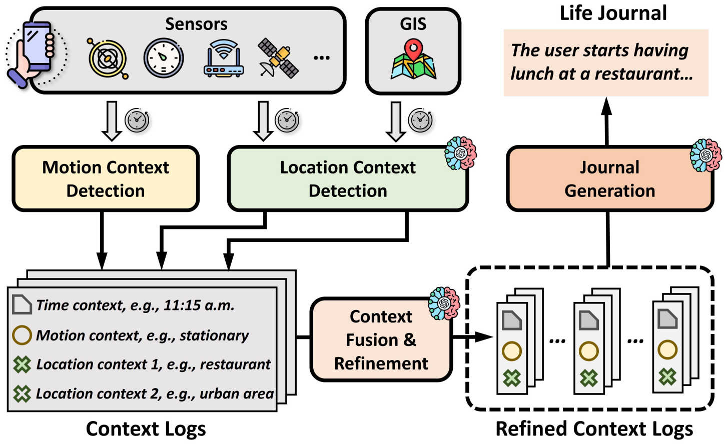


Fig. 3. AutoLife overview.

contexts as flexible texts and utilizes another LLM-based module to enhance their precision and reduce text length. Finally, AutoLife aggregates the enhanced context logs over a long duration and processes them through the *journal generation* module, where LLMs synthesize the information to generate comprehensive life journals for users.

3.3 Input Sensors

It is intuitive that any single sensor data, e.g., the accelerometer or GPS location, cannot provide sufficient information to infer accurate journals. Therefore, our system integrates data from multiple sensors. Below is an overview of the chosen sensor features and how they are pre-processed.

- **Accelerometer** sensors capture the device's accelerations. We use step-count algorithms [17] to estimate the user's steps from a duration of accelerometer readings, which serves as another important indicator.
- **Gyroscope** measures the device's angular velocity, which can be integrated with the accelerometer to estimate device orientation. The human-caused acceleration [16] is also an important feature, which can be computed by fusing the two sensors.
- **Barometer** measures air pressure, which can be used to estimate rough altitude using the barometric formula [2]. We then compute the altitude change over a time period as $\Delta h = h_i - h_j$, where h_i represents the altitude at time i. The altitude change is a valuable feature for detecting user movement.
- **GPS speed** reflects the user's movement on the horizontal plane. Since satellite signals may be blocked when the user is indoors, the speed reported by the localization module can be unreliable. We filter GPS speed data when the number of detected satellites is fewer than 5.
- **GPS location** provides the geographic coordinates, consisting of latitude and longitude. Similarly, GPS data can be unreliable indoors and we filter out locations where the horizontal accuracy radius, as reported by the Android API [4], exceeds 50 meters.
- **WiFi** signals can also help determine the user's location and are used for localization in the Google Fused Location Provider [3]. Recent studies [48, 72] have shown that WiFi Service Set Identifiers (SSIDs) can offer valuable insights into a user's surroundings.

Note that during implementation, we access the geographic location from Andoird Fused Location Provider API [3], which fuses multiple sources including GPS and WiFi for more accurate localization.

Algorithm 1: Motion detection algorithm in AutoLife.

Input: step count s per minute, acceleration excluding gravity a m/s^2, altitude change Δh m, horizontal speed v m/s.
Output: motion list L.

```
L ← [];
if s <= 2 & a <= 0.1 & |Δh|<= 0.1 & v <= 0.1 then
    L ← L + [‘stationary’];
else if s <= 10 & |Δh|<= 1.0 & v < 0.5 then
    L ← L + [‘limited motion’];
end
if s >= 140 & 2.0 <= v <= 5.0 then
    L ← L + [‘jogging/running’];
if s >= 50 & v < 1.8 then
    L ← L + [‘walking’];
if s >= 50 & v >= 4.0 then
    L ← L + [‘cycling’];
if (s <= 5 & v > 2) | v > 5 then
    L ← L + [‘vehicle/subway/ferry/train’];
if s <= 10 & Δh > 2.5 & v < 2 then
    L ← L + [‘escalator/elevator’];
end
return L;
```

4 CONTEXT DETECTION

This section will elaborate on how we fuse the input sensors and derive motion or location contexts for life journaling.

4.1 Motion Context

Motion information like walking is a key indicator for determining users' behaviors. Extensive research in HAR [23, 30, 32, 33, 43, 72–74, 85] has demonstrated the potential of leveraging motion sensors to identify activities like jogging or cycling. However, these approaches cannot be directly applied to life journaling because most available public datasets [43, 61, 62, 80] cover only a limited range of sensor modalities, users, devices, and labeled data, making it challenging to build general models for recognizing activities.

To build a general solution, we propose a new rule-based motion detection algorithm by exploiting multimodal sensors. As outlined in Algorithm 1, our approach fuses multiple features post-process by raw sensor data, including step counts, acceleration excluding gravity, altitude change, and GPS horizontal speed. The rules are based on commonsense knowledge; for example, if the step count is low while the speed is high, the user is likely using transportation. Despite leveraging multiple sources, ambiguities still arise when determining certain activities, so our algorithm acknowledges the limitations of sensors and can output multiple possible motions when the input data is inconclusive. Later, we leverage LLMs to reduce these ambiguities by incorporating location context.

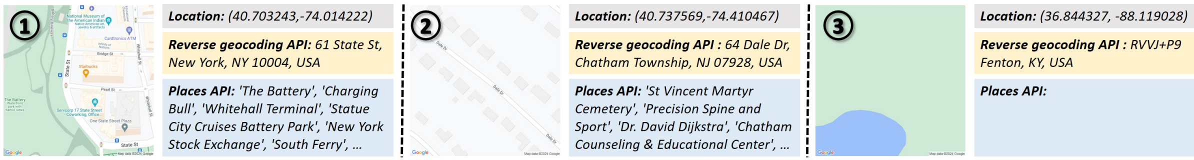


Fig. 4. Examples of detecting location contexts with address and places. Results are from Google Maps Geocoding and Places API, respectively. The left side shows the map segments centered at corresponding locations.

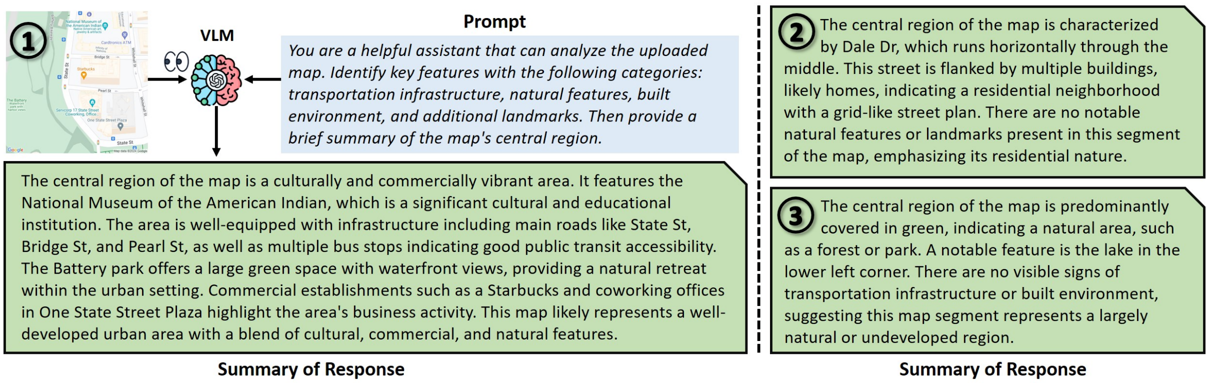


Fig. 5. Examples of detecting location contexts by analyzing map images with VLM. The results are generated from GPT-4o [52] and input images are the maps in Fig. 4.

We reference thresholds in gait and activity analysis studies [9, 66, 67] and fine-tune them using our dataset. We evaluate the proposed algorithm with SVM and neural network on our dataset (detailed in Section 7.1) and Sussex-Huawei Locomotion Dataset [21]. Please refer to Section 7.4 for more details.

4.2 Location Context

Location context is also crucial for accurately inferring a user's activity. However, detecting location contexts using ubiquitous sensors on smartphones is not straightforward. In this section, we design a low-cost solution for detecting location contexts.

4.2.1 Location Context from GPS location. Modern smartphones can easily access geographic locations, including latitude and longitude, through their positioning modules. However, GPS locations often do not provide sufficient information on their own. Our first idea is to exploit these locations with the existing Geographic Information Systems (GIS) like Google Maps [27] or OpenStreetMap [53], which offer comprehensive details about places worldwide and are widely used in daily life. However, identifying the location contexts from existing GIS is non-trivial. We first explore two available APIs of these GIS platforms:

- **Reverse Geocoding API** [26, 49]: This API converts geographic coordinates into addresses, providing a basic level of location context, such as 'South Ferry, New York, NY 10004'.
- **Places API** [25]: This API generates a list of nearby places within a specified radius around a geographic coordinate. It is important to note that there is a maximum limit on the number of place results, such as 20 for the Google Maps Places API [25].

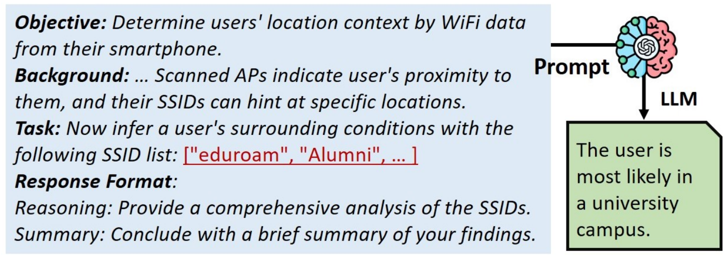


Fig. 6. Location context detection with WiFi SSID. The red underlined texts in the prompt would be replaced by the scanned WiFi SSIDs.

We illustrate three example results of the above methods in Fig. 4, which cover different scenarios, including public, residential, and recreational areas. The addresses obtained from the Reverse Geocoding API do not convey informative location context. While the Places API can provide extensive landmarks information in urban areas like the 'New York Stock Exchange' and 'Charging Bull', it has limitations. In residential areas, as shown in the second example, the Places API tends to be biased toward public places, such as sports or educational centers, and may not accurately reflect the residential context. Furthermore, when the device is in a suburban area, both APIs may fail to return any relevant context. In summary, these two methods are not universally effective for location context detection across all scenarios.

To address this challenge, we observe that map segments on the other hand can provide more general and stable information than address or place texts. A map itself is an image where shapes, colors, and patterns all convey significant contextual information. For instance, the grey rectangles in the second case of Fig. 4 likely represent houses, while the blue area in the third case indicates a body of water. More importantly, map segments are widely available and can be easily accessed through services like the Google Maps Static API [24]. Therefore, we propose analyzing map images to derive more comprehensive location contexts.

However, interpreting maps is challenging, as it requires extensive knowledge to understand the shapes, colors, and texts presented in the images. Inspired by the rapid progress and success of recent vision language models (VLMs) [11, 19, 36, 52, 70], we propose leveraging existing VLMs to analyze map images without any additional training. As shown in Fig. 5, we use GPT-4o [52] to detect location contexts from the three maps in Fig. 4. The results demonstrate GPT-4o's strong zero-shot ability to extract key features from the maps and generate accurate contexts for all three cases. Thus, modern VLMs offer a new and reliable approach to identifying location contexts from maps.

We use the Google Static Map API [24] to retrieve map images, configuring three key parameters: the central location of the map (specified by geographic coordinates from the positioning module), the image size (500×500 pixels), and the zoom level (18), which ensures the map covers a sufficient area encompassing approximately 250×250 m^2 [24]. To avoid redundant API calls for maps with close centers, we implement a grid system with a size of 100×100 m^2 and all coordinates in the same grid share the map image and location contexts. Additionally, since map information is generally stable, we maintain a key-value database to store the location contexts generated by VLMs. The key is a string representing the grid location, while the value is a string containing the location context. This approach allows us to reuse the inference results from VLMs, further reducing costs.

4.2.2 Location Context from WiFi SSID. In addition to GPS locations, WiFi Service Set Identifiers (SSIDs) can also provide valuable location context [48, 72]. For example, if a smartphone detects an SSID containing 'Starbucks', it suggests that the user is near or inside a Starbucks. However, analyzing SSIDs requires a substantial amount of commonsense knowledge to interpret the names of various places, including restaurants, transportation hubs,

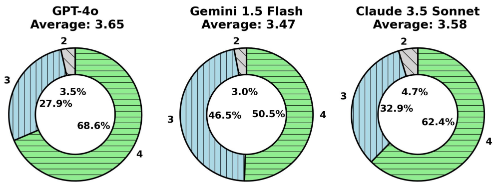


Fig. 7. Performance of VLMs on location context detection with maps. The higher the scores, the better the performance.

Table 1. Performance of LLMs on location context detection with WiFi SSIDs.

Metric	GPT-3.5	Gemini 1.5 Flash	Claude 3 Sonnet
Score (↑)	3.51	3.43	3.25
Win Rate (↑)	**42.2%**	31.8%	26.0%
Recall (↑)	0.928	0.962	**0.997**
Specificity (↑)	**0.895**	0.842	0.789

landmarks, and more. To address this, we adapt the approach from [72] and utilize LLMs, such as ChatGPT [50], to derive location contexts from WiFi SSIDs as shown in Fig. 6. We observe that many WiFi access points in public networks share identical SSIDs, such as 'eduroam'. To optimize token usage, we preprocess the SSID list by removing duplicate SSIDs.

4.3 Location Context Evaluation

We conduct two experiments to evaluate the performance of existing commercial LLMs/VLMs in location context detection. The data collection process is detailed in Section 6. We find these tasks are special as analyzing maps or WiFi SSIDs requires a broad base of general knowledge, an area where existing LLMs may often outperform humans [51]. To assess their performance, we evaluate the models by judging or rating their responses. We recruited 18 volunteers and collected a total of 330 and 360 scores for the two tasks, respectively.

In the first task of map interpretation using VLMs, we evaluate the performance of GPT-4o (`gpt-4o-2024-05-13`) [52], Gemini Flash (`gemini-1.5-flash`) [64], and Claude 3 Sonnet (`claude-3-5-sonnet-20240620`) [6]. We instruct the VLMs to generate descriptions for maps and designed a questionnaire to rate these descriptions. Each question included one map image, a description generated by an LLM, and four rating options ranging from 1 to 4, where '1' indicates "The description mismatches the map" and '4' represents "The description well matches the map". The questions were randomly sampled from 300 instances of map segments in Hong Kong, and the models were anonymized to the volunteers.

Fig. 7 presents the overall scores of the three VLMs that demonstrate impressive performance in this task, which requires interpreting shapes and texts (both in English and Chinese). The average scores were high, with GPT-4o, Gemini Flash, and Claude 3 Sonnet achieving 3.68, 3.47, and 3.58, respectively. Notably, none of the models hallucinates and receives a score of 1, underscoring the feasibility of using VLMs to interpret maps for location context detection.

The second task, location context detection using WiFi SSIDs, is considerably more challenging for humans, as SSIDs often contain diverse and unfamiliar text, such as restaurant, company, or place names. We conducted 50 tests where volunteers rated the performance of LLMs on a scale from 1 to 4, with the assistance of ground-truth location context. For the remaining 310 tests, we had the LLMs compete against each other, asking volunteers to select the best response among. We also introduced two additional options: "SSIDs are not informative"—when SSIDs lack unique identifiers for detailed location contexts, and "Not sure"—when the models gives similar responses or when the SSIDs were particularly difficult to analyze. Since this task involves only processing text inputs, we replaced GPT-4o (`gpt-4o-2024-05-13`) with lighter-weight GPT-3.5 (`gpt-3.5-turbo-0125`).

Table 1 presents the performance of the three models across 360 tests. In this task, recall refers to the ratio of instances where the LLMs successfully generate valid context relative to the instances where volunteers consider SSIDs to be informative. Specificity represents the ratio of instances where LLMs generate valid context relative to the instances where volunteers believe SSIDs lack location indicators. Win rates indicate the number of cases

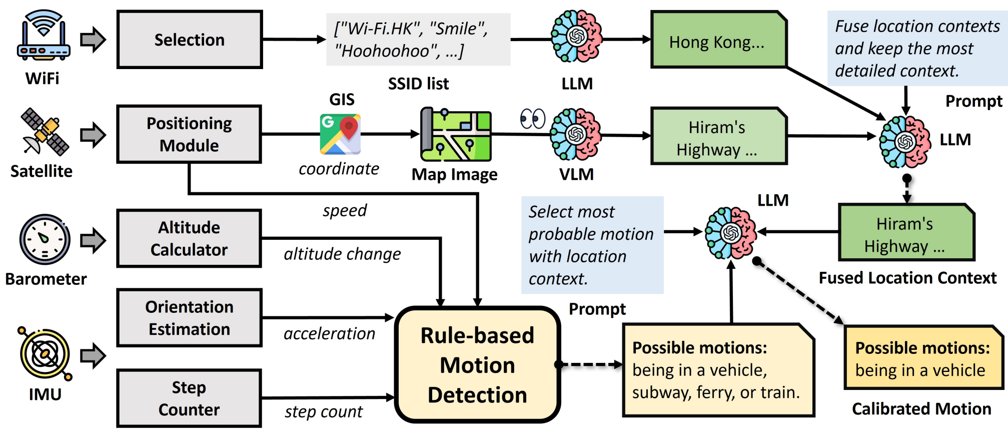


Fig. 8. Location and motion context fusion in AutoLife.

in which each model beats the other two. Overall, all models achieve good performance, demonstrating that using them to analyze SSIDs for location context detection is effective.

5 CONTEXT FUSION

Now we have explored how to detect users' contexts with various sensors and this section will elaborate on how these contexts can be fused to enhance precision.

Both map-based and SSID-based methods can provide valuable location contexts; however, we observe they have distinct features:

- **Map-based location context** is effective in almost all situations but tends to provide only general descriptions, such as identifying an area as commercial or residential. Additionally, it struggles to offer detailed information in public areas with numerous points of interest (POIs); for example, it may not determine which specific store a user is in within a shopping mall.
- **SSID-based location context** can be fine-grained in some cases, such as identifying specific restaurants or campuses. However, it becomes less effective in suburban areas with few WiFi access points or when scanned SSIDs are not informative, such as 'Redmi 9A' or 'SjFaHJ6echEs,' which lack identifiers that can be used to derive meaningful location contexts.

5.1 Location Context Fusion

Therefore, we propose fusing the two location contexts to obtain the most fine-grained context. Since both contexts are represented as text and the fusion task requires extensive commonsense knowledge, we believe LLMs are well-suited for this task. The upper part of Fig. 8 illustrates the workflow for location context detection. The LLM is prompted to merge the location contexts and retain the most detailed and specific information—in the shown case, "Hiram's Highway", as nearby SSIDs are not highly informative. If the user is in an urban area, the SSID-derived context can provide valuable information, such as identifying a restaurant by an SSID like "McDonald's". This approach allows us to generate the most detailed and fine-grained location contexts based on multiple smartphone sensor signals.

5.2 Motion Calibration

With the location context, actually we can further improve the accuracy of motion contexts, especially when our rule-based method provides multiple possible options. For instance, if a user is detected at a high GPS speed, determining the exact transportation mode can be challenging. But if we know the user is on a water surface, it's likely they are on a ferry. To achieve this, we propose calibrating the detected motion types using location context.

This task also requires a significant amount of commonsense knowledge, making LLMs an effective solution. We represent both the location and motion contexts as text and use LLMs to calibrate the motions, as illustrated in Fig. 8. The LLM is prompted to "select the most probable motion given the location context". For example, if the primary location context is "Hiram's Highway", the transportation mode is likely to be "being in a vehicle". This approach allows us to further remove the ambiguity of motions and enhance the precision of motion contexts.

6 LIFE JOURNALING

The previous section details how to obtain accurate contexts from sensor data, though this process is limited to short time windows, e.g., 15 seconds. But generating a life journal requires processing sensor data over much longer durations like hours. This section explains how to aggregate contexts from extended time windows and generate life journals.

6.1 Context Refinement

To get long-term context information, we should aggregate context logs over time. However, simply combining these contexts as texts can result in overly lengthy and less accurate data. To address this, we apply several optimizations to the context fusion process.

First, we observe that location contexts from neighboring time windows may vary in quality or detail. For example, one context might describe "a restaurant", while the context from the neighboring window can specify "a McDonald's restaurant", with the latter providing more information. Therefore, we also need to fuse location contexts over time. Additionally, as shown in Fig. 5, the location contexts generated by LLMs are often lengthy. For instance, the token size of the location context in Fig. 5 case 1 is 131 tokens for ChatGPT. Directly aggregating this length of context over an hour would result in a text with a token size of 7,860 if we detect map-based location context every minute. To reduce the text length, we introduce a simple yet effective instruction in the prompt when fusing location contexts derived from maps and SSIDs, i.e., "present concise location logs" or "present concise motion logs".

Combining all the designs, we organize multiple location contexts from neighboring time windows in the format of "*[time-1](map location context, WiFi location context), ..., [time-n](map location context, WiFi location context)*" to further incorporate time context, where n is set to 15 in AutoLife. The LLM is instructed to perform three steps: 1) "select the most detailed location context from two contexts at the same time", 2) "select the most specific and detailed location context across time", and 3) "present the enhanced and concise location logs as *[time-1](fused location context), ..., [time-n](fused location context)*". The refined location contexts are extracted and then used to calibrate the motion contexts as outlined in Fig. 8.

6.2 Journal Generation

Now we can combine these refined contexts to cover longer durations like hours. We organize the three contexts over time as "*[time-1](calibrated motion context, fused location context), ..., [time-n](calibrated motion context, fused location context)*". Similarly, we believe that the task of deriving a journal from a list of contexts is well-suited for LLMs, as it requires a substantial amount of common sense knowledge. As shown in Fig. 9, we provide the LLMs with a prompt that instructs them to analyze the context logs and infer high-level semantic activities like dining.

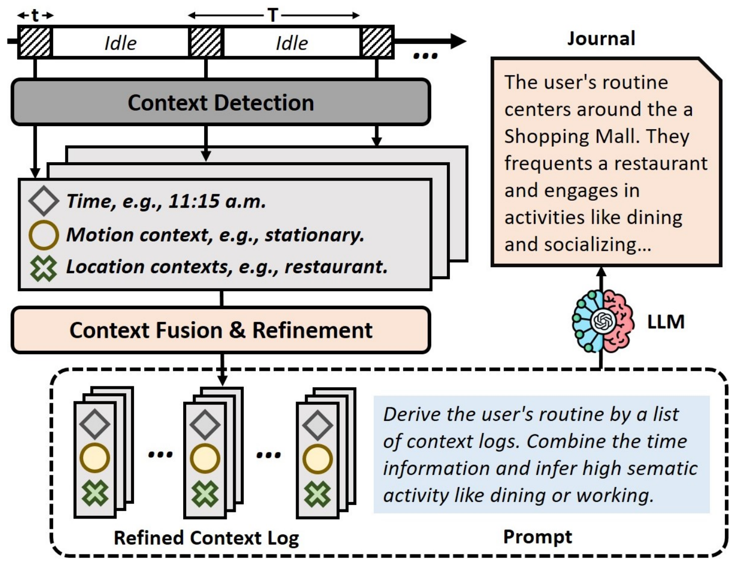


Fig. 9. Journal generation in AutoLife.

To improve the journal quality and control the format of the generated journals, we also include several example journal entries in the prompt, such as, "In the morning, the user spends time at a local library, likely reading and researching".

We also observed that many LLMs, like ChatGPT, tend to include some " comments" on the response, such as "The routine consists of a blend of work and leisure", which do not align with our goal of generating factual journals. To address this, we use another LLM session with the prompt -"remove any subjective comments if they exist" to further polish the journal. This process yields the final journal for the user, summarizing their behaviors over a long duration.

6.3 Data Collection Duty Cycle

Although life journaling requires long durations of sensor data, it is unnecessary for our system to continuously and consistently collect data from smartphones, such as scanning WiFi signals for hours, as this would consume excessive energy [18]. Therefore, we design a duty cycle for the data collection, as shown at the top of Fig. 9. The system periodically activates the collection process and then enters an idle state for a while. The context detection module then processes the collected sensor data to generate contexts. The parameters t and T represent the collection duration and period, respectively. To allow sufficient time for the smartphone to scan WiFi and compute a more accurate step count, we set t to 15 seconds. The collection period T is set to 60 seconds and its impact will be evaluated in Section 7.4.

7 EVALUATION

7.1 Implementation

APP design. Since life journaling is a novel application, to the best of our knowledge, there is no existing dataset available for it. Therefore, we develop an Android application that runs a foreground service to regularly access sensor data, such as satellite and WiFi signals, from the system APIs. The data collection process follows the duty cycle described in Section 6.3, with all sensor data being implicitly saved in files for offline processing.

Table 2. Life journaling dataset summary.

Category	Number	Description
Participants	10	7 males, 3 females; weight: 40–70 kg; height: 160–185 cm; age: 24–33
Devices	11	Samsung Galaxy S8, Galaxy A55, Galaxy S24; OPPO Find X8, Reno 10; HUAWEI Pura 70, nova 12s, Mate 40 RS; vivo V30; Google Pixel 7; Xiaomi 14
Experiments	100	263.6 hours total, average duration: 2.25 hours
City	4	Hong Kong, Shenzhen, Zhuhai, Chengdu
Places	-	Shopping mall, hospital, campus, restaurant, park, cinema, hotel, railway station, airport, sea coast, and more
Activity	-	Walking, running, reading, using elevators, taking the subway, studying, dining, hiking, and more

Dataset. We recruit 10 volunteers from China to collect an extensive dataset in various scenarios, and Table 2 presents a summary of its key characteristics. Our project is reviewed and approved by our Institutional Review Board (IRB) under a Human Research Ethics Protocol. Informed consent was obtained from all participants involved before the data collection. Appropriate safeguards were implemented to anonymize and securely store the collected data. Furthermore, all participants agreed to share their anonymized data publicly.

Each volunteer was provided with clear instructions regarding the data collection process: they were given an experimental smartphone, instructed to activate data collection within the designated application, and then carried out their daily activities as usual while using the device. The smartphone was not required to be tightly attached to the volunteers; for example, they were free to place the phone on a table while having a meal. We collect data from 100 experiments, totaling about 263.6 hours, with an average experiment duration of about 2.25 hours — significantly longer than the sensing durations targeted in HAR studies, e.g., typically a few seconds. For each experiment, the volunteer provides two similar and concise text descriptions of their behaviors, referred to as the *reference journals*, for evaluation purposes.

Models settings. AutoLife has many LLM-based modules, and there are numerous potential combinations of available models. We establish a default configuration with several representative LLMs. In the location context module, we select GPT-4o (`gpt-4o-2024-05-13`) [52] for map interpretation and GPT-3.5 (`gpt-3.5-turbo-0125`) [50] for SSID interpretation. In the context fusion and journal generation modules, we adopt GPT-4o mini (`gpt-4o-mini-2024-07-18`). We will compare the performance of some representative LLMs on key modules.

Prompts. All prompts include a specified response format ("reasoning" and "summary" [71]) to constrain LLMs, and every response undergoes a keyword detection process to extract the key "summary" content, such as location context or journal entries. We illustrate key parts of the prompts in Fig. 5, 8, and 9.

Metrics. To evaluate the quality of journals generated by AutoLife, we measure the similarities between them and the reference journals using chrF [55] and BERTScore [82], both of which are widely adopted metrics in the natural language processing domain. We also define LLMs as hallucinating if they do not follow the specified response format and the target context cannot be extracted from their responses.

Baseline. To comprehensively evaluate AutoLife, we also establish a baseline solution, referred to as *SenLLM* in this paper, which simply aggregates raw sensor across time and inputs them into LLMs for journal generation.

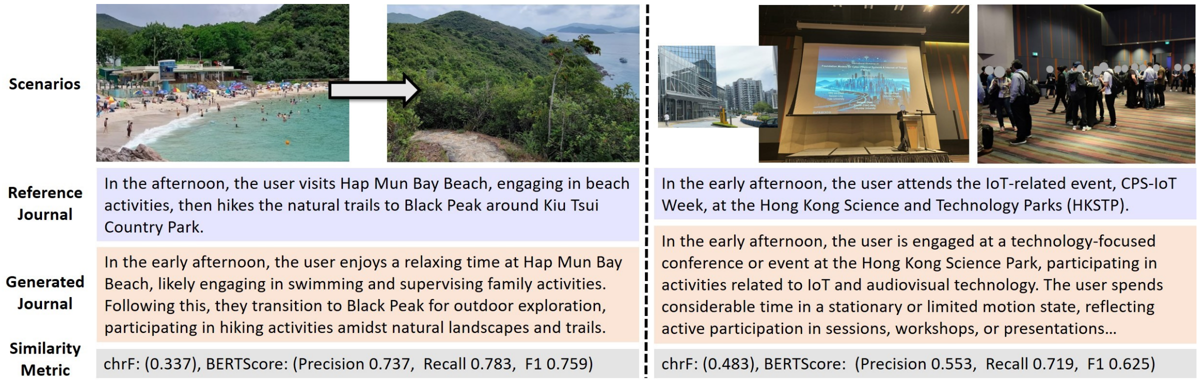


Fig. 10. Life journal examples generated by AutoLife with GPT-4o mini.

Table 3. Performance of AutoLife and baseline methods. The 'P', 'R', and 'F1' represent precision, recall, and F1 score, respectively. All numbers are average value among all experiments.

Generation LLM	Method	Hallucination rate (↓)	chrF (↑)	BERTScore (↑)		
				Precision	Recall	F1-Score
GPT-4o	SenLLM	8%	0.412	0.531	0.622	0.570
	AutoLife	1%	0.489	0.582	0.737	0.642
GPT-4o mini	SenLLM	0%	0.324	0.492	0.535	0.511
	AutoLife	1%	**0.514**	0.602	0.722	0.651
Claude 3 Opus	SenLLM	6%	0.377	0.551	0.629	0.584
	AutoLife	0%	0.498	**0.603**	**0.745**	**0.662**
Gemini 1.5 Pro	SenLLM	7%	0.351	0.528	0.597	0.558
	AutoLife	0%	0.437	0.596	0.685	0.635
Llama3 70B*	SenLLM	17%	0.352	0.501	0.577	0.534
	AutoLife	8%	0.449	0.591	0.676	0.628

7.2 Main Results

Fig. 10 shows two example journals generated by AutoLife together with ground-truth scenario photos and reference journals. In the first case, the user visits a beach and then goes hiking. The AutoLife successfully captures the key activity like 'hiking' and location context like the name of the beach. Similarly, the generated journal also demonstrates high quality in the second case and AutoLife derives the user attending a conference or event. Interestingly, it derives that the event is IoT-related from a scanned SSID "CPS-IoT WEEK 2024". Overall, the generated journal aligns well with the reference journal and their similarities achieve high BERTScore.

Interestingly, we observe that LLMs sometimes give some complementary descriptions like "participation in sessions", which are valid but do not appear in the reference journals. Additionally, LLM can make some reasonable speculations based on the motion and location contexts, e.g., 'swimming', which would decrease the precision. Both factors typically result in the generated journal being longer than the reference journal, which causes the recall to be higher than the precision.

To provide a comprehensive quantitative evaluation, we test AutoLife and the baseline solution SensorLLM using different LLMs for journal generation, including GPT-4o [52], Claude 3 [5], Gemini 1.5 [58], and Llama3 [1]. Table 3 presents their overall performance across various metrics. We observe that when inputting raw sensor data, many LLMs can capture partial insights about users' behaviors (mainly from WiFi SSIDs). However, all LLMs show higher hallucination rates and achieve lower scores (as indicated by the SensorLLM). In contrast, LLMs integrated with AutoLife achieve significantly better results; for example, Claude 3 Opus achieves high BERTScore precision and recall of 0.601 and 0.745, respectively. Notably, only a few models hallucinate during the challenging task of life journaling with AutoLife, and the much lighter-weight and open-source model like Llama3 70B also performs well. Overall, the results clearly demonstrate the superior effectiveness of AutoLife over the baseline solution.

7.3 Impact of Time Period

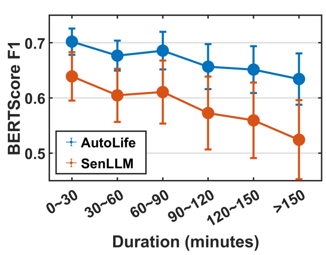


Fig. 11. Performance comparison between AutoLife and SenLLM under different durations.

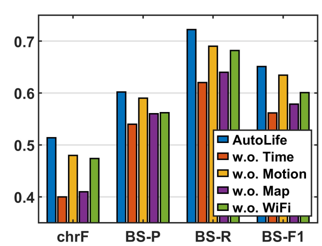


Fig. 12. AutoLife performance under different contexts. The 'BS' indicates BERTScore.

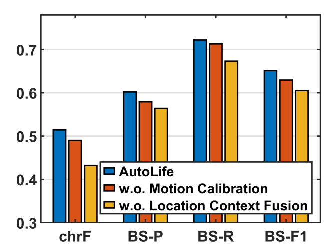


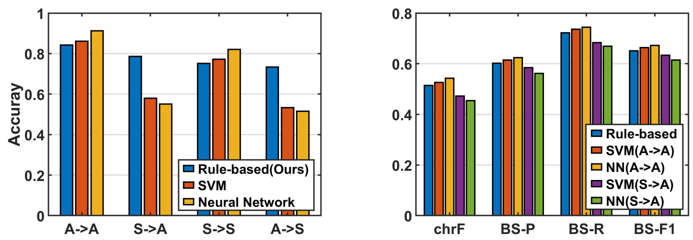
Fig. 13. AutoLife performance under different modules. The 'BS' indicates BERTScore.

We then examine the impact of experiment duration on AutoLife, and Fig. 11 presents the average results of all experiments for three representative LLMs (GPT-4o, Claude 3 Opus, Gemini 1.5 pro) across different durations, ranging from 0–30 minutes to over 150 minutes. As the duration increases, the LLMs maintain better performance compared with the baseline solution. Notably, AutoLife achieves average scores higher than 0.62, even for durations exceeding 180 minutes. These results indicate that AutoLife is less sensitive to duration compared to the baseline, thanks to its multi-layer framework. However, extended durations can still affect system performance. We further discuss the impact of duration in Section 9.

7.4 Ablation Study

In the ablation study, we adopt the default model settings introduced in Section 7.1 and investigate the impacts of several key technical designs.

Impact of resources. We first investigate the impact of different data sources on journal generation. Fig. 12 shows the quality of journals generated using various combinations of resources. For example, 'w.o. motion' indicates that only the location context was used for journal generation. The results demonstrate that combining all available resources—including both motion and location contexts (map-based and WiFi-based)—yields the best performance for AutoLife. We observe that the time and map location contexts play key roles in journal quality. Removing it resulted in higher decreases for all matrices, which justifies our design of incorporating additional time and location contexts.



(a) Motion detection comparison.

(b) Varying detection methods.

Fig. 14. Impact of motion context. The 'A' and 'S' indicate our dataset and the Sussex-Huawei dataset. The 'A->S' means the models are trained on our dataset and tested on the Sussex-Huawei dataset. The 'BS' indicates BERTScore.

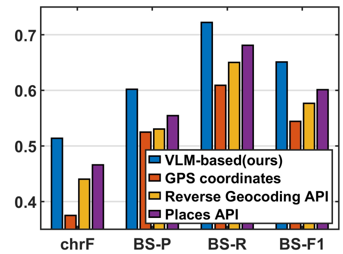


Fig. 15. AutoLife performance under different location contexts. Both API related methods are based on Google Maps. The 'BS' indicates BERTScore.

Impact of context fusion. We also evaluate the performance of AutoLife without LLM-based location context fusion (Section 5.1) or LLM-based motion calibration (Section 5.2). As shown in Fig. 13, comparing AutoLife with these two alternatives reveals that both contribute to improvements across the four metrics. For instance, omitting LLM-based location context fusion leads to a large decrease in chrF. These results show the effectiveness of LLM-based context fusion and enhanced contexts can benefit the downstream journal generation task.

Impact of context detection methods. We then examine the performance of different motion detection methods, incorporating both motion and location context derived from GPS data. For motion detection, we compare our proposed rule-based algorithm with two machine learning models: a Support Vector Machine (SVM) and a neural network (three layers with hidden sizes of 32 and 64). We evaluate these methods on both our dataset and the Sussex-Huawei Locomotion Dataset [21]. To further assess their generalizability, we adopt a cross-dataset evaluation protocol commonly used in prior works [30, 57, 73], and define four settings: 1) Training and testing on our dataset, 2) Training and testing on the Sussex-Huawei Locomotion Dataset, 3) Training on our dataset and testing on the Sussex-Huawei dataset, and 4) Training on the Sussex-Huawei dataset and testing on our dataset. Certain labels like 'car' and 'bus' in the Sussex-Huawei dataset are merged. The number of samples is 831 in our dataset and 13,544 in the Sussex-Huawei dataset, with 90% used for training and 10% for testing. The neural network is trained with 100 epochs with a batch size of 32. As shown in Fig. 14(a), while SVM and neural networks can slightly outperform our rule-based method when trained and tested on the same dataset, they experience significant performance degradation when transferred across datasets due to the overfitting issues. We also evaluate the impact of these motion detection methods on journal generation, and observe similar trends, as shown in Fig. 14(b). Therefore, we believe the rule-based approach is more generalizable for our task, as many motion types can be reliably inferred using commonsense knowledge, and large-scale, high-quality training data is unavailable.

We then examine how different GPS-based location context detection methods perform within our system. Specifically, we compare our VLM-based approach with the following alternatives: 1) GPS coordinates only, 2) Google Reverse Geocoding API, 3) Google Places API. Each method follows the same downstream pipeline, including motion calibration, before generating life journals. As shown in Fig. 15, our VLM-based approach consistently outperforms the baselines, demonstrating the added value of leveraging map images and vision-language understanding to extract richer and more accurate location contexts.

Impact of context fusion models. Different from Table 3, this experiment focuses on the impact of LLMs on context fusion (Section 5). As shown in Fig. 16, we test three representative LLMs for both fusing map- and WiFi-based location contexts and calibrating motions using the fused location context. Although these models are not high-end LLMs, they still achieve fair performance, demonstrating the effectiveness of AutoLife's task decomposition, which allows LLMs to handle each subtask effectively. These results, along with those in Table 3 confirm the generalizability of AutoLife across different LLMs.

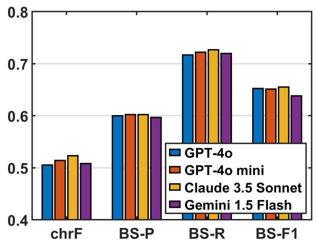


Fig. 16. Performance comparison with different fusion models. The 'BS' indicates BERTScore.

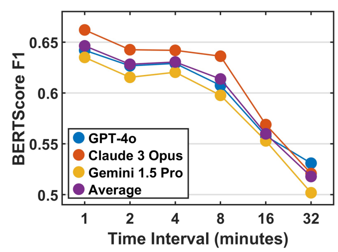


Fig. 17. Performance comparison with different sampling intervals from 1 minute to 32 minutes.

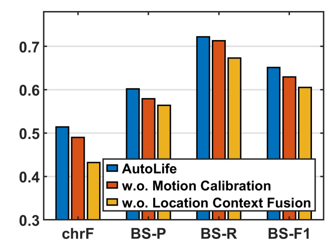


Fig. 18. Performance comparison with different prompt design. The 'BS' indicates BERTScore.

Impact of sampling interval As detailed in Section 6.3, we designed a data collection duty cycle where the application periodically collects data from the smartphone. This experiment evaluates the impact of the sampling interval on the quality of generated journals. As shown in Fig. 17, the results show that all three models achieve stable and high BERTScore F1 when the intervals range from 1 to 8 minutes. However, when the interval increases over 16 minutes, the overall performance degrades significantly across all models. While a higher sampling interval reduces system overhead, such as power consumption and token usage, it also leads to information loss and lower-quality journals, making it a trade-off parameter.

Impact of prompt. In Fig. 18, we examine how two key designs in prompts impact journal qualities by removing the "present concise description" instruction for context refinement (Section 6.1) or the journal examples for the journal generation (Section 6.2). The 'concise' instruction provides a slight overall performance improvement while significantly reducing token usage, which will be discussed in the next subsection. Including journal examples, however, contributes to a more substantial improvement in overall performance.

7.5 System Cost

Token cost. We first evaluate the token cost of AutoLife using GPT-4o mini and pricing as of August 2024. The frequency of journal generation and cleaning (removing "subject" comments introduced in Section 6.2) is set to once per hour and all token usages are the averages across all experiments. As shown in Table 4, the total cost is $\$3.2 \times 10^{-2}$ per hour. Additionally, by adopting a map-based location context database, map contexts can be reused and the token usage can be reduced by 82%, lowering the total cost to $\$2.2 \times 10^{-2}$ per hour. The token usages with the 'concise' instruction are reduced by 5.1%, 7.8%, and 9.0% for the outputs of location fusion, motion calibration, and the input for journal generation, respectively. Overall, the system cost is affordable using commercial LLMs, which can be further reduced by leveraging open-source models like Llama 3.

Power consumption. We compare the power consumption of our journaling app with that of other commonly used applications to better understand its overhead on several commercial smartphones. In all experiments, the smartphone screen remains on to ensure consistent testing conditions. As shown in Table 5, our app does not introduce significant power overhead and consumes less power than several widely used cases.

Table 4. Token usage summary of AutoLife. The token usage includes both input and output token numbers.

Module	Token Usage	Freq.	Price (dollar)
Map Context	437, 316	1/min	1.5×10^{-2}/hr
WiFi Context	309, 335	1/min	1.5×10^{-2}/hr
Location Fusion	1236, 611	4/hr	5.9×10^{-4}/hr
Motion Calibration	602, 395	4/hr	3.5×10^{-4}/hr
Journal Generation	2015, 394	1/hr	5.4×10^{-4}/hr
Journal Clearning	98, 60	1/hr	5.1×10^{-5}/hr
Total	-	-	3.2×10^{-2}/hr

Table 5. Hourly battery drain percentage comparison with screen on.

Usage Scenario	**Pixel 7**	**Redmi Note 11T Pro**	**Xiaomi 14**	**Xiaomi 15**	**Samsung S24**	**Samsung A55**	**Avg.**
Idle	2.8	2.0	2.6	1.3	1.9	3.1	2.3
Music Listening (Netease Cloud)	3.0	2.3	2.6	2.5	2.1	2.5	2.5
Data Collection (AutoLife)	7.0	6.8	7.6	8.4	6.4	9.8	7.7
Navigation (Google Map)	9.8	8.0	9.4	8.8	18.4	16.0	11.7
Running Record (KEEP)	11.3	11.6	13.6	14.2	16.5	12.0	13.2

7.6 User Study

We also conduct a user study experiment to evaluate how the generated journals met the quality standards expected by users using five key metrics as follows: (1) **Clarity** is assessed by examining how easy the journal is to understand and whether the information is presented logically and coherently. (2) **Conciseness** evaluates whether the journal conveys its message efficiently, avoiding redundant information. (3) **Correctness** focuses on the accuracy of the content, measuring that the information presented is factual and error-free. (4) **Completeness** ensures that the journal thoroughly covers all relevant aspects, providing the necessary detail without omitting relevant information. (5) **Relevance** assesses the degree to which the content is focused on important and meaningful aspects.

Six volunteers rated each metric for randomly sampled 20 experiments (excluding the experiments where the model hallucinates). They refer to the reference journals and rate the generated journals on a four-point scale, where 1 indicates "completely does not meet the criteria", and 4 indicates "completely meets the criteria". Fig. 19 shows the average scores of three models rated by volunteers, with AutoLife achieving much higher scores higher across all metrics, significantly outperforming SensorLLM in terms of correctness, completeness, and relevance. These results further validate the effectiveness and usability of AutoLife.

8 REALWORLD APPLICATION

Putting AutoLife into real-world application, we find that it can be naturally integrated into digital journaling apps. Therefore, we develop an automatic journaling APP based on AutoLife, namely **AutoJournal**. Unlike

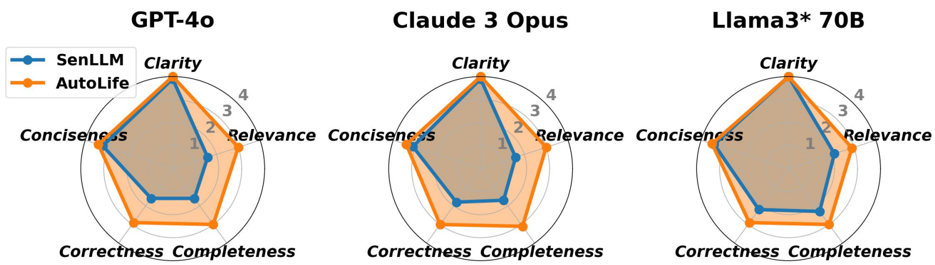


Fig. 19. User study results for AutoLife.

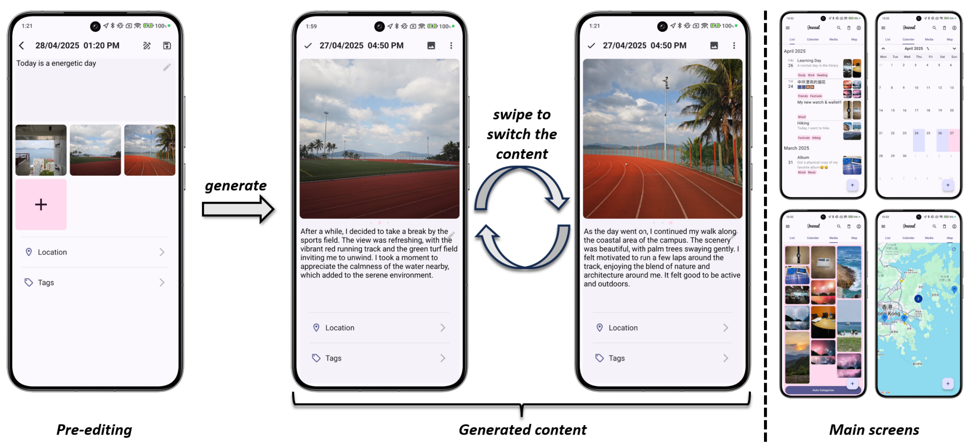


Fig. 20. Editing screens (left) and main screens (right) of Life Journaling.

existing apps such as Day One [38] or Journal [31], AutoJournal integrates AutoLife to enable automatic journal generation.

8.1 Core Features and UI

AutoJournal presents the following key features:

- **Journal generation:** As shown in the left part of Fig. 20, users can select several photos, optionally type a draft journal, choose a location, and select a journal style (defaulting to "storytelling"). The app then generates a comprehensive journal by combining information from photos, sensors, and other available sources.
- **Journal display**: The app supports multiple display modes for individual journals. As shown in the middle part of Fig. 20, journals are carefully partitioned and tightly bound to photos. Users can swipe between different photos to view the corresponding journal content. Alternatively, they can browse a mixed mode where photos and text are integrated, sliding vertically to explore different sections.
- **Journal management**: As shown in the right part of Fig. 20, the app offers a variety of presentation methods for all journals, including list view, calendar view, an overview of photos directing journals, and

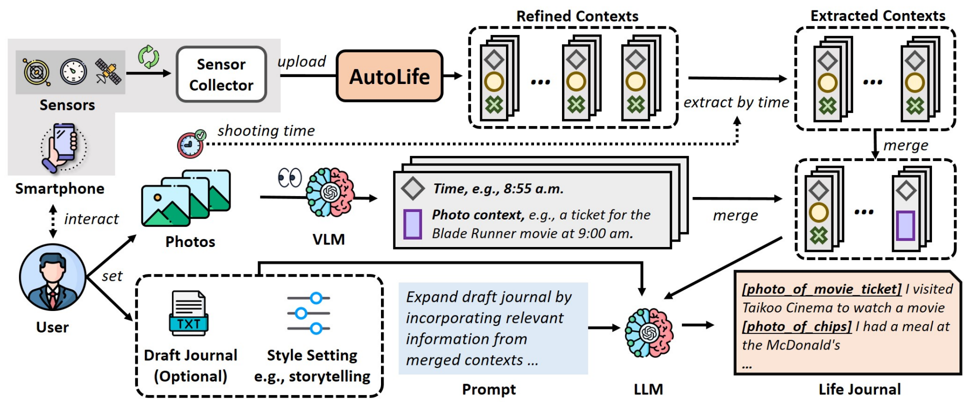


Fig. 21. AutoLife-based journal generation framework for AutoJournal.

a map-based view clustering journal entries by location. Users are free to edit, rewrite, or delete journal entries.

- **Others:** The app also supports user management and journal synchronization to the cloud, enabling multi-device access and offline editing to ensure optimal usability under different network conditions.

8.2 System Design

Our system consists of two main components: an Android application and a cloud server. The Android application features a user-friendly interface that supports basic journaling functions, allowing users to view, add, edit, and delete journal entries. The app continuously collects sensor data from mobile devices and uploads it to the cloud server. A local database is integrated to enable offline storage of journal entries and user information. The cloud server provides APIs that interact with the mobile application to collect and manage sensor, journal, and user data. It implements key algorithms integrated with AutoLife to enable automatic journal generation. A multi-tenant database architecture is adopted to securely store and manage data across different users.

Algorithm design: For a journaling application, we observe that photos are widely used and serve as key carriers of users' memories. Building on this idea, we propose fusing photo information with sensor-derived contexts to enable more accurate and comprehensive journal generation. To address the challenge of integrating diverse information sources, including sensor contexts and photos, we further present a framework based on AutoLife, as illustrated in Fig. 21.

Users can select several photos (up to 9) from their system album and optionally input a draft journal, which can be very concise. They can also optionally specify content focus and text styles, with default settings of "recording" and "storytelling," respectively. Once the user triggers the journal generation service, the selected photos, draft journal, and style preferences are uploaded to the cloud server. Particularly, the sensor collector continuously and periodically collects sensor data in the background and uploads them to the cloud server.

As shown in Fig. 21, the server-side framework processes the received sensor data to extract contexts using AutoLife, and those contexts are saved as files for possible future journal generation. Once the cloud server receives a generation request, it extracts the relevant contexts based on the earliest and latest shooting timestamps of the selected photos, with a 15-minute margin added to both ends. Simultaneously, all photos are processed through VLMs to generate textual descriptions of their visual content. These photo descriptions are then aligned and merged with the sensor-derived contexts according to their timestamps. Finally, the merged information, along with any user-provided draft and style settings, is fed into an LLM to generate the final journal entry.

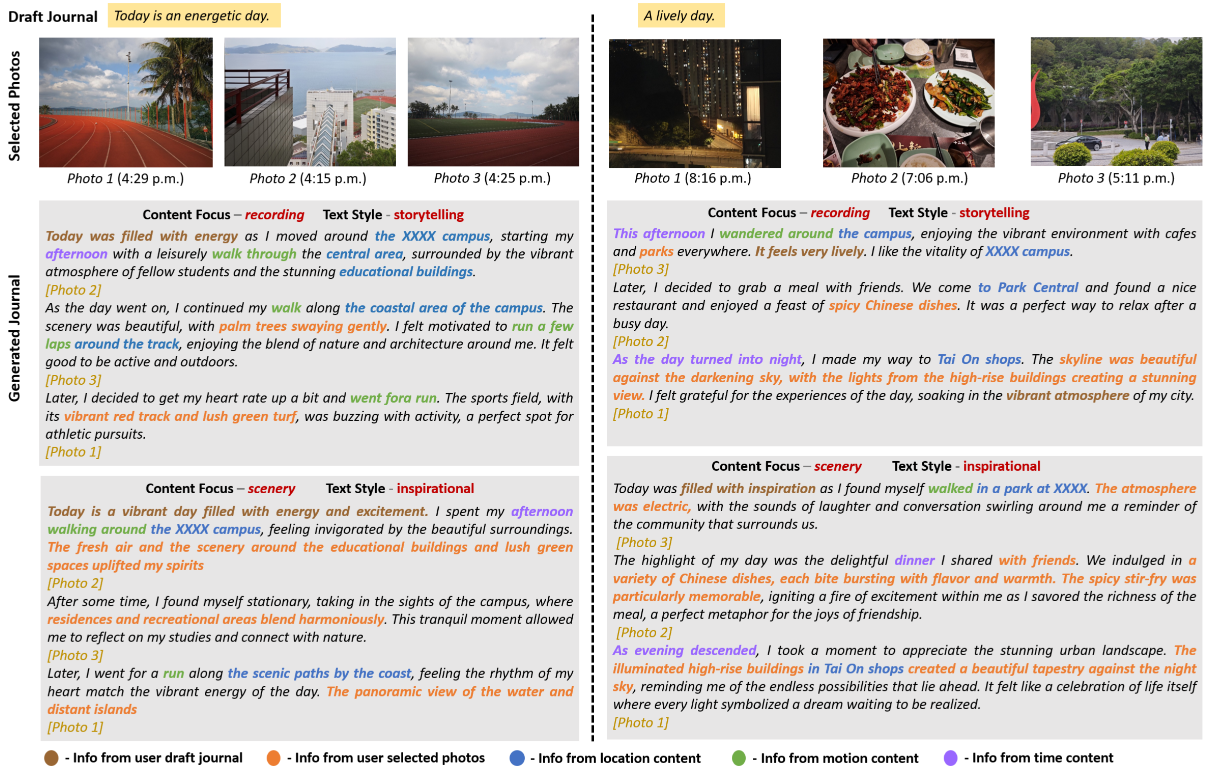


Fig. 22. Journal examples generated by our journaling app.

Journal style: To tailor journal styles to different user needs, we allow users to select from various writing options. For example, a business trip journal might favor a straightforward itinerary format, while a travel journal could emphasize personal reflections or detailed image descriptions. By offering different preset prompts, as shown in Fig. 21, we can guide the LLM to adapt its writing style by adjusting the emphasis on user-input text, images, and sensor-based location data. Our app allows users to customize journal generation through two style parameters: focus point and text style. Focus point defines the primary content emphasis of the journal, while text style determines the linguistic tone and format of the journal.

Implementation: Our application is developed for the Android platform. The UI is built using the Compose framework and follows Google's Material Design 3 guidelines to ensure a modern and intuitive user experience. For local data storage, we use SQLite within the mobile application. The cloud server is implemented using the Django framework, with a cloud-based MongoDB database deployed to manage all sensor data uploaded by clients. Additionally, we synchronize users' basic data, such as journal content, directly to the cloud through Firebase services. For AI capabilities, we adopt GPT-4o (`gpt-4o-2024-05-13`) [52] for both the VLM and LLM components.

8.3 Use Cases

Figure 22 presents two example journals generated by our app. Each case includes the user's draft journal (if provided), selected photos, and chosen writing style. In the visualization, different portions of the generated journal are highlighted using different colors to indicate their information sources—such as photos, motion, or location contexts.

In the first example, between 4 and 5 p.m., the user began at the main teaching building on campus, strolled leisurely through the student dormitories, and eventually arrived at the sports field, where they walked several laps and ran. The generated journal effectively captures the user's major activities and manages to reorganize the photos over time. Importantly, all sources of information—sensor data and photos—contribute meaningfully to the final journal, demonstrating the value of fusing multiple contexts.

In addition, journal styles can be customized by selecting different writing modes. For instance, with the default style, the journal uses past tense and transition words like "later" and "continued," highlighting the passage of time and changes in activity. When the user selects the "scenery"-focused style, the journal contains more descriptive content drawn from the photos, emphasizing visual elements and surroundings.

Similarly, our system generates high-quality journal entries for the second example as well. These results collectively demonstrate the effectiveness and robustness of our overall design.

8.4 User Study

We also conduct a user study for the AutoJournal app. We invite 5 volunteers, each of whom uses the app over several days to generate a total of 7 journals, resulting in 35 journals covering 188.68 hours (with an average duration of 5.38 hours per journal). The dataset also includes 76 photos provided by users. We adopt the same five evaluation metrics used in the previous user study for AutoLife (Section 7.6): clarity, conciseness, correctness, completeness, and Relevance. As shown in Fig. 23, AutoJournal achieves average scores of 3.67, 2.79, 2.97, 3.27, and 2.70 for the respective metrics. Notably, both correctness and completeness are close to 3.0, indicating a reasonable level of factual and contextual coverage. Compared with the results in Section 7.6, we observe a slight decrease in clarity and conciseness when incorporating photo information. While the inclusion of images enriches the contextual information, it also increases complexity. The LLM may tend to include excessive detail, which presents a direction for future improvement. Overall, AutoJournal achieves fairly good performance.

We survey the volunteers and collect several representative comments as shown in Fig. 24. Most users express excitement and find it impressive that the app could generate journals automatically. However, some raise privacy concerns, while others suggest additional features, such as support for more media types and customizable writing styles.

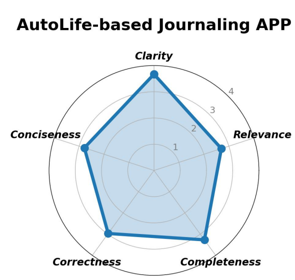


Fig. 23. User study results of AutoJournal app.

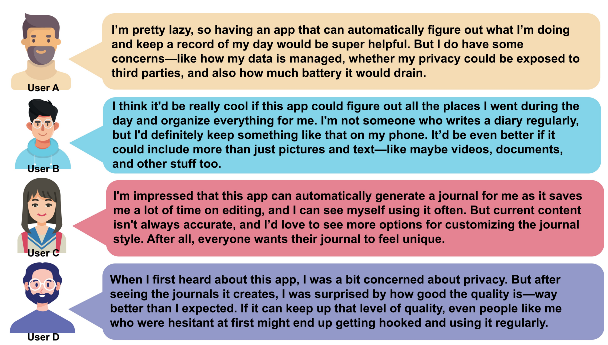


Fig. 24. Representative comments from AutoJournal users.

In addition, we asked the 5 participants to evaluate several popular journaling apps — including Day One [38], Journey [42], and Journal (Apple) [31] — using a five-level rating scale across four aspects: UI friendliness, media

APP	UI friendliness	Media Support	Platform Support	AI-related Features
AutoJournal (Ours)	★★★☆ 1. Support flexible journal editing and generation. 2. Support diverse forms of journal presentation.	★★☆☆ **Main Content**: rich text, image **Appendices**: time, location, journal type tags	★☆☆☆ Andriod	★★★★ 1. Automatic journal generation 2. Support diverse writing styles 3. Automatic photo classification
Day One	★★★☆ 1. Support flexible journal editing and generation. 2. Support diverse forms of journal presentation.	★★★★ **Main Content**: rich text, Image, drawing, video, PDF **Appendices**: Time, location, weather, journal type tags	★★★★ Andriod, iOS, Windows, MacOS, Web	☆☆☆☆
Journey	★★★☆ 1. Support flexible journal editing and generation. 2. Support diverse forms of journal presentation.	★★★☆ **Main Content:** rich text, image, drawing **Appendices**: time, location, weather, action, mood, journal type tags	★★★★ Andriod, iOS, Windows, MacOS, Linux	★★☆☆ 1. Guided journal generation with conversations. 2. Sample Q&A about past journals (e.g. what have I done in yyyy/mm/dd?, how many pictures have I take?)
Journal (Apple)	★★★★ 1. Support flexible journal editing and generation. 2. Support diverse forms of journal presentation. 3. Smooth animation	★★★★ **Main Content:** Rich Text, Image, drawing, audio **Appendices:** time, location, weather, action, mood, recent music.	★☆☆☆ iOS	★★★☆ 1. Journal rewrite to different styles e.g. concise 2. Guided journal generation with recommended photos, and location.

Fig. 25. Journaling application comparison.

support, platform support, and AI-related features. A rating of 0 indicates very low satisfaction, while 4 indicates the highest level of satisfaction. As shown in Fig. 25, all users gave AutoJournal high scores for its AI capabilities, expressing appreciation for its automatic journal generation. However, due to its current limitations in media and platform support, it received lower scores in those categories. Expanding media compatibility and cross-platform availability are important directions for our future development.

All these comments and feedback are valuable for us to further improve AutoJournal, and we also discuss several of these points in Section 9.

9 DISCUSSION

Privacy Concerns: Life journaling inherently involves handling sensitive data such as GPS location and WiFi SSID, which can raise privacy concerns. We adopt several strategies to mitigate these risks. (1) Our app implements the data collection module using an Android foreground service [15], which displays a persistent notification to ensure users are aware of and can monitor ongoing data collection. (2) Users are given full control over both the collected sensor data and all generated outputs, including contexts and journals, to enhance transparency and trust. (3) Data is anonymized before being sent to cloud-based model APIs. In future work, we can develop small language models tailored for context processing and journal generation that can run entirely on-device, thereby offering enhanced privacy protection.

Trade-off of motion label granularity: Our current motion detection algorithm supports a limited set of motion labels, selected based on those that can be reliably detected using smartphone sensors. Increasing the granularity of motion labels — for example, distinguishing specific activities like eating — would inevitably lead to higher detection errors. This limitation stems from two main factors: (1) the inherent constraints of smartphones, which are not tightly attached to the body, and (2) the limited sensing capabilities of the onboard sensors. Overall, this represents a trade-off between label precision and detection reliability. One potential solution is to integrate data from additional wearable devices, such as smartwatches or earbuds, which could provide richer motion information and enable finer-grained activity recognition. However, this approach introduces additional cost and complexity for users. Future work should also explore practical and user-friendly designs for multi-device sensor fusion to address this challenge effectively.

Practical deployment challenge: One major challenge for deployment is ensuring continuous sensor data collection. We evaluated the performance of our data collection module implemented as a foreground service on six Android devices. The results show that the module remains active for extended periods (over 10 hours), even when the screen is locked or when other resource-intensive apps (e.g., games) are running in the foreground, demonstrating the practicality of our design. But we find certain devices (e.g., Xiaomi models) exhibit delayed sampling intervals (e.g., extending to 15 minutes) when the screen remains locked for a couple of minutes (e.g., 20 minutes). The sampling rate returns to normal once the device becomes active again. While such throttling can lead to temporary data loss and negatively impact the quality of generated journals, it is worth noting that prolonged idle periods themselves can serve as meaningful indicators of user activities, such as resting or focused work. In addition, frequent user interactions with smartphones in real-world scenarios help mitigate the impact of these idle states. Incorporating additional wearable devices (e.g., smartwatches) could also help address this issue by providing complementary sensing data during idle periods.

Limitations: One limitation of AutoLife is that it currently struggles to generate high-quality journals for extended time periods (e.g., over 6 hours). One possible way to address this issue is to partition the sensor data into shorter windows, e.g., a few hours, and then combine the journals generated from these individual segments. We observe that when a user remains in the same location for a long duration, the system may produce redundant context segments that can degrade journal quality and unnecessarily increase processing costs. Therefore, designing an effective redundancy removal mechanism is essential. Another limitation is that AutoLife currently focuses on generating natural, factual descriptions of users' daily activities. However, users may also wish to include emotional context — such as feeling happy or sad — in their journals. Capturing such subjective states inevitably requires some form of user input, which falls outside the scope of our current fully automated design.

Future works: Beyond incorporating new sensors and on-device models, future work includes enhancing the system with additional sensors — such as ambient light or temperature — which could provide more contextual clues and enrich activity inference. We would like to explore other potential applications of AutoLife. For instance, we can generate time-use summaries (e.g., "You spent 3 hours commuting and 5 hours working today") to help users better understand and manage their daily routines. By analyzing journaling data over time, we can also build richer user profiles for personalized recommendations, such as activities, services, or product suggestions tailored to individual preferences.

10 CONCLUSION

In this paper, we propose a novel mobile sensing application called *life journaling* and design an automatic life journaling system AutoLife utilizing ubiquitous smartphones. To accurately derive a user's journal, AutoLife exploits multiple contexts and extensive common knowledge within LLMs. We collect a dataset and establish a benchmark to evaluate the quality of life journals. Experiment results show that AutoLife can generate high-quality journals, and a journaling app is developed based on AutoLife that can incorporate photo information for advanced journal generation. We believe that life journaling represents a significant milestone application by integrating LLMs with sensor data, paving the way for new applications in personal daily life tracking and beyond.

ACKNOWLEDGMENTS

We thank all reviewers for their insightful comments. This work is supported by the Global STEM Professorship Scheme of Hong Kong, the HKUST start up grant, and the Research Grants Council (RGC) General Research Fund (GRF) 16210425. Mo Li is the corresponding author.

REFERENCES

[1] Meta AI. 2024. Meta LLaMA 3: Advancing Open-Source AI. https://ai.meta.com/blog/meta-llama-3/

[2] David G Andrews. 2010. *An introduction to atmospheric physics*. Cambridge University Press.

[3] Android Developers. 2024. Fused Location Provider. https://developers.google.com/location-context/fused-location-provider.

[4] Android Developers. 2024. Location. https://developer.android.com/reference/android/location/Location.

[5] Anthropic. 2024. Claude 3 Model Card. https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf

[6] Anthropic. 2024. Introducing the Claude 3 Family. https://www.anthropic.com/news/claude-3-family.

[7] Riku Arakawa, Jill Fain Lehman, and Mayank Goel. 2024. Prism-q&a: Step-aware voice assistant on a smartwatch enabled by multimodal procedure tracking and large language models. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 4 (2024), 1–26.

[8] Oresti Banos, Juan-Manuel Galvez, Miguel Damas, Hector Pomares, and Ignacio Rojas. 2014. Window size impact in human activity recognition. *Sensors* 14, 4 (2014), 6474–6499.

[9] Richard W Bohannon. 1997. Comfortable and maximum walking speed of adults aged 20—79 years: reference values and determinants. *Age and ageing* 26, 1 (1997), 15–19.

[10] Marc Bolanos, Mariella Dimiccoli, and Petia Radeva. 2016. Toward storytelling from visual lifelogging: An overview. *IEEE Transactions on Human-Machine Systems* 47, 1 (2016), 77–90.

[11] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. 2023. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818* (2023).

[12] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020).

[13] Wenqiang Chen, Jiaxuan Cheng, Leyao Wang, Wei Zhao, and Wojciech Matusik. 2024. Sensor2Text: Enabling Natural Language Interactions for Daily Activity Tracking Using Wearable Sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 4 (2024), 1–26.

[14] Wikipedia contributors. 2024. Lifelog. https://en.wikipedia.org/wiki/Lifelog.

[15] Android Developers. 2024. Foreground Services. https://developer.android.com/develop/background-work/services/foreground-services

[16] Android Developers. 2024. Motion sensors. https://developer.android.com/develop/sensors-and-location/sensors/sensors_motion#sensors-motion-linear.

[17] Android Developers. 2024. Step Counter Sensors. https://developer.android.com/develop/sensors-and-location/sensors/sensors_motion#sensors-motion-stepcounter.

[18] Android Developers. 2024. Wi-Fi Scanning | Android Developers. https://developer.android.com/develop/connectivity/wifi/wifi-scan.

[19] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. 2023. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378* (2023).

[20] Jim Gemmell, Gordon Bell, and Roger Lueder. 2006. MyLifeBits: a personal database for everything. *Commun. ACM* 49, 1 (2006), 88–95.

[21] Hristijan Gjoreski, Mathias Ciliberto, Lin Wang, Francisco Javier Ordonez Morales, Sami Mekki, Stefan Valentin, and Daniel Roggen. 2018. The university of sussex-huawei locomotion and transportation dataset for multimodal analytics with mobile devices. *IEEE Access* 6 (2018), 42592–42604.

[22] Glogger. 2013. LifeGlogging cameras 1998 2004 2006 2013 labeled. https://commons.wikimedia.org/wiki/File:LifeGlogging_cameras_1998_2004_2006_2013_labeled.jpg.

[23] Taesik Gong, Yeonsu Kim, Jinwoo Shin, and Sung-Ju Lee. 2019. Metasense: few-shot adaptation to untrained conditions in deep mobile sensing. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 110–123.

[24] Google. 2024. Google Maps Static API Overview. https://developers.google.com/maps/documentation/maps-static/overview.

[25] Google. 2024. Place Search. https://developers.google.com/maps/documentation/places/web-service/nearby-search. Accessed: [insert date here].

[26] Google. 2024. Reverse Geocoding. https://developers.google.com/maps/documentation/geocoding/requests-reverse-geocoding.

[27] Google LLC. 2024. Google Maps. https://maps.google.com.

[28] Cathal Gurrin, Hideo Joho, Frank Hopfgartner, Liting Zhou, and Rami Albatal. 2016. Ntcir lifelog: The first test collection for lifelog research. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 705–708.

[29] Cathal Gurrin, Alan F Smeaton, Aiden R Doherty, et al. 2014. Lifelogging: Personal big data. *Foundations and Trends® in information retrieval* 8, 1 (2014), 1–125.

[30] Zhiqing Hong, Zelong Li, Shuxin Zhong, Wenjun Lyu, Haotian Wang, Yi Ding, Tian He, and Desheng Zhang. 2024. CrossHAR: Generalizing Cross-dataset Human Activity Recognition via Hierarchical Self-Supervised Pretraining. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 2 (2024), 1–26.

[31] Apple Inc. 2023. Apple launches Journal app, a new app for reflecting on everyday moments. https://www.apple.com/hk/en/newsroom/2023/12/apple-launches-journal-app-a-new-app-for-reflecting-on-everyday-moments/.

[32] Yash Jain, Chi Ian Tang, Chulhong Min, Fahim Kawsar, and Akhil Mathur. 2022. ColloSSL: Collaborative Self-Supervised Learning for Human Activity Recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 1 (2022), 1–28.

[33] Sijie Ji, Xinzhe Zheng, and Chenshu Wu. 2024. HARGPT: Are LLMs Zero-Shot Human Activity Recognizers? *arXiv preprint arXiv:2403.02727* (2024).

[34] Shiqi Jiang, Zhenjiang Li, Pengfei Zhou, and Mo Li. 2019. Memento: An emotion-driven lifelogging system with wearables. *ACM Transactions on Sensor Networks (TOSN)* 15, 1 (2019), 1–23.

[35] Wenjun Jiang, Chenglin Miao, Fenglong Ma, Shuochao Yao, Yaqing Wang, Ye Yuan, Hongfei Xue, Chen Song, Xin Ma, Dimitrios Koutsonikolas, et al. 2018. Towards environment independent device free human activity recognition. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. 289–304.

[36] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. 2022. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094* (2022).

[37] Amel Ksibi, Ala Saleh D Alluhaidan, Amina Salhi, and Sahar A El-Rahman. 2021. Overview of lifelogging: current challenges and advances. *IEEE Access* 9 (2021), 62630–62641.

[38] Day One Journal App | Your Journal For Life. 2024. Automattic. https://dayoneapp.com/.

[39] Shengzhong Liu, Shuochao Yao, Jinyang Li, Dongxin Liu, Tianshi Wang, Huajie Shao, and Tarek Abdelzaher. 2020. GIobalFusion: A Global Attentional Deep Learning Framework for Multisensor Information Fusion. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–27.

[40] Wei Liu, Xue Li, and Daoli Huang. 2011. A survey on context awareness. In *2011 International Conference on Computer Science and Service System (CSSS)*. IEEE, 144–147.

[41] Xin Liu, Daniel McDuff, Geza Kovacs, Isaac Galatzer-Levy, Jacob Sunshine, Jiening Zhan, Ming-Zher Poh, Shun Liao, Paolo Di Achille, and Shwetak Patel. 2023. Large Language Models are Few-Shot Health Learners. *arXiv preprint arXiv:2305.15525* (2023).

[42] Two App Studio Pte. Ltd. 2024. Journey. https://journey.cloud/.

[43] Mohammad Malekzadeh, Richard G Clegg, Andrea Cavallaro, and Hamed Haddadi. 2019. Mobile sensor data anonymization. In *Proceedings of the international conference on internet of things design and implementation*. 49–58.

[44] Microsoft. 2024. SenseCam Research Project. https://www.microsoft.com/en-us/research/project/sensecam/?from=https://research.microsoft.com/en-us/um/cambridge/projects/sensecam/&type=exact.

[45] Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. 2023. Large Language Models as General Pattern Machines. In *Proceedings of the 7th Conference on Robot Learning (CoRL)*.

[46] Subigya Nepal, Arvind Pillai, William Campbell, Talie Massachi, Eunsol Soul Choi, Xuhai Xu, Joanna Kuc, Jeremy F Huckins, Jason Holden, Colin Depp, et al. 2024. Contextual AI Journaling: Integrating LLM and Time Series Behavioral Sensing Technology to Promote Self-Reflection and Well-being using the MindScape App. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. 1–8.

[47] Subigya Nepal, Arvind Pillai, William Campbell, Talie Massachi, Michael V Heinz, Ashmita Kunwar, Eunsol Soul Choi, Xuhai Xu, Joanna Kuc, Jeremy F Huckins, et al. 2024. MindScape Study: Integrating LLM and Behavioral Sensing for Personalized AI-Driven Journaling Experiences. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies* 8, 4 (2024), 1–44.

[48] Jiazhi Ni, Fusang Zhang, Jie Xiong, Qiang Huang, Zhaoxin Chang, Junqi Ma, BinBin Xie, Pengsen Wang, Guangyu Bian, Xin Li, et al. 2022. Experience: Pushing indoor localization from laboratory to the wild. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*. 147–157.

[49] Nominatim developer community. 2024. Reverse Geocoding. https://nominatim.org/release-docs/develop/api/Reverse/.

[50] OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[51] OpenAI. 2023. Weak to Strong Generalization. https://openai.com/index/weak-to-strong-generalization/.

[52] OpenAI. 2024. Hello GPT-4o. https://openai.com/index/hello-gpt-4o/.

[53] OpenStreetMap contributors. 2024. OpenStreetMap. https://www.openstreetmap.org.

[54] Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. 2023. Kosmos-2: Grounding Multimodal Large Language Models to the World. *arXiv preprint arXiv:2306.14824* (2023).

[55] Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the tenth workshop on statistical machine translation*. 392–395.
[56] Ronald Poppe. 2010. A survey on vision-based human action recognition. *Image and vision computing* 28, 6 (2010), 976–990.
[57] Xin Qin, Yiqiang Chen, Jindong Wang, and Chaohui Yu. 2019. Cross-dataset activity recognition via adaptive spatial-temporal transfer learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 4 (2019), 1–25.
[58] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530* (2024).
[59] Aaqib Saeed, Tanir Ozcelebi, and Johan Lukkien. 2019. Multi-task self-supervised learning for human activity detection. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 2 (2019), 1–30.
[60] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100* (2022).
[61] Muhammad Shoaib, Stephan Bosch, Ozlem Durmaz Incel, Hans Scholten, and Paul JM Havinga. 2014. Fusion of smartphone motion sensors for physical activity recognition. *Sensors* 14, 6 (2014), 10146–10176.
[62] Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. 2015. Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition. In *Proceedings of the 13th ACM conference on embedded networked sensor systems*. 127–140.
[63] Pejman Tahmasebi, Serveh Kamrava, Tao Bai, and Muhammad Sahimi. 2020. Machine learning in geo-and environmental sciences: From small to large scale. *Advances in Water Resources* 142 (2020), 103619.
[64] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805* (2023).
[65] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).
[66] Catrine Tudor-Locke, Susan B Sisson, Tracy Collova, Sarah M Lee, and Pamela D Swan. 2005. Pedometer-determined step count guidelines for classifying walking intensity in a young ostensibly healthy population. *Canadian journal of applied physiology* 30, 6 (2005), 666–676.
[67] Salvatore Vanini, Francesca Faraci, Alan Ferrari, and Silvia Giordano. 2016. Using barometric pressure data to recognize vertical displacement activities on smartphones. *Computer Communications* 87 (2016), 37–48.
[68] Michalis Vrigkas, Christophoros Nikou, and Ioannis A Kakadiaris. 2015. A review of human activity recognition methods. *Frontiers in Robotics and AI* 2 (2015), 28.
[69] Limin Wang, Sheng Guo, Weilin Huang, and Yu Qiao. 2015. Places205-vggnet models for scene recognition. *arXiv preprint arXiv:1508.01667* (2015).
[70] Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, et al. 2023. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *arXiv preprint arXiv:2305.11175* (2023).
[71] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.
[72] Huatao Xu, Liying Han, Mo Li, and Mani Srivastava. 2023. Penetrative ai: Making llms comprehend the physical world. *arXiv preprint arXiv:2310.09605* (2023).
[73] Huatao Xu, Pengfei Zhou, Rui Tan, and Mo Li. 2023. Practically Adopting Human Activity Recognition. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. 1–15.
[74] Huatao Xu, Pengfei Zhou, Rui Tan, Mo Li, and Guobin Shen. 2021. LIMU-BERT: Unleashing the Potential of Unlabeled Data for IMU Sensing Applications. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 220–233.
[75] Bufang Yang, Siyang Jiang, Lilin Xu, Kaiwei Liu, Hai Li, Guoliang Xing, Hongkai Chen, Xiaofan Jiang, and Zhenyu Yan. 2024. Drhouse: An llm-empowered diagnostic reasoning system through harnessing outcomes from sensor data and expert knowledge. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 4 (2024), 1–29.
[76] Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. 2023. mplug-owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178* (2023).
[77] Yinggang Yu, Dong Wang, Run Zhao, and Qian Zhang. 2019. RFID based real-time recognition of ongoing gesture with adversarial learning. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 298–310.
[78] Özgür Yürür, Chi Harold Liu, Zhengguo Sheng, Victor CM Leung, Wilfrido Moreno, and Kin K Leung. 2014. Context-awareness for mobile sensing: A survey and future directions. *IEEE Communications Surveys & Tutorials* 18, 1 (2014), 68–93.
[79] Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414* (2022).

[80] Mi Zhang and Alexander A Sawchuk. 2012. USC-HAD: A daily activity dataset for ubiquitous activity recognition using wearable sensors. In *Proceedings of the 2012 ACM conference on ubiquitous computing*. 1036–1043.
[81] Shugang Zhang, Zhiqiang Wei, Jie Nie, Lei Huang, Shuang Wang, and Zhen Li. 2017. A review on human activity recognition using vision-based method. *Journal of healthcare engineering* 2017, 1 (2017), 3090343.
[82] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations*.
[83] Mingmin Zhao, Shichao Yue, Dina Katabi, Tommi S Jaakkola, and Matt T Bianchi. 2017. Learning sleep stages from radio signals: A conditional adversarial architecture. In *International Conference on Machine Learning*. PMLR, 4100–4109.
[84] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. 2014. Learning deep features for scene recognition using places database. *Advances in neural information processing systems* 27 (2014).
[85] Zhijun Zhou, Yingtian Zhang, Xiaojing Yu, Panlong Yang, Xiang-Yang Li, Jing Zhao, and Hao Zhou. 2020. XHAR: Deep Domain Adaptation for Human Activity Recognition with Smart Devices. In *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 1–9.