Mobile Lifelogger - recording, indexing, and understanding a mobile user's life

Snehal Chennuru, Peng-Wen Chen, Jiang Zhu, and Ying Zhang

Carnegie Mellon University
Moffett Field, CA 94035
{snehal.chennuru, pengwen.chen, jiang.zhu, joy.zhang}@sv.cmu.edu

Abstract. Lifelog system involves capturing personal experiences in the form of digital multimedia during an entire lifespan. Recent advancements in mobile sensor technologies have helped to develop these systems using commercial smart phones. These systems have the potential to act as a secondary memory and also aid people who struggle with episodic memory impairment (EMI). Despite their huge potential, there are major challenges that need to be addressed to make them useful. One of them is how to index the inherently large lifelog data so that the person can efficiently retrieve the log segments that interest him / her most. In this paper, we present an ongoing research of using mobile phones to record and index lifelogs using activity language. By converting sensory data such as accelerometer and GPS readings into activity language, we are able to apply statistical natural language processing techniques to index, recognize, segment, cluster, retrieve, and infer high-level semantic meanings of the collected lifelogs. Based on this indexing approach, our lifelog system supports easy retrieval of log segments representing past similar activities and automatic lifelog segmentation for efficient browsing and activity summarization.

Key words: Lifelogger, activity language, mobile computing, indexing heterogenous data

1 Introduction

Memory, our ability to store, retain and recall information, is crucial to day to day life. But, our memory fades and we tend to forget intricate details of our experiences. Memory-related problems can be very serious for people who suffer from brain injuries or have memory diseases like the Alzheimer's. As of September 2009, more than 35 million people around the world are living with Alzheimer's disease for which episodic memory impairment (EMI) is the main symptom [12]. And according to [3], the prevalence of Alzheimer's is thought to reach approximately 107 million people by 2050.

In 1945, Vannevar Bush proposed a prototype computer system named MEMEX, whose main functionality is to share people's burden in memorizing things. Such system has great potential in a variety of applications and is

particularly useful for people who suffered from EMI [10]. To assist people with this ever growing population, Bush's MEMEX concept seems to be a promising solution and thus gives birth to the personal lifelog research area. To fulfill Bush's vision, a personal lifelog system must be able to 1) store a large volume of personal multimedia data and 2) efficiently retrieve the relevant data based on users' requests.

Technologies today make it possible to capture one's life experience in digital format. The advancement in mobile sensors and ubiquitous computing allows lifeLog systems to record almost every aspects of one's life to provide a digital memory [6]. While recording and storing all sensor information in a database poses some engineering challenges, indexing them is the key to make lifelog system useful. We need the index so that we can retrieve important pieces of memory from the lifelogs. We can not expect users to annotate everything in the lifelog and automatic extracting semantics from images, audio and video is still an open research problem. Without a convenient method to index and retrieve the recorded data, such logged memory is of little use.

In this paper, we present a new approach to indexing lifelog data using the activity language. In this ongoing research, we convert the ambulatory sensor inputs such as accelerometers' readings into the so called *activity language* and use the activity language as the main index of the multimedia lifelogs. The activity language approach enables us to use statistical natural language processing methods to index, retrieve, cluster, and summarize lifelogs which are not easy or not possible for images, audio, and video information.

The rest of the paper is organized as follows. We first describe our lifeLogger system in Section 2. Section 3 introduces the concept of "Activity as Language" where we quantize the sensory input and convert it into a text representation in order to interpret the meaning of lifelogs. In Section 3.2, 3.3 and 3.5, we present algorithms and preliminary results on activity recognition, similar activity retrieval, and automatic lifelog segmentation. Finally, we conclude our findings and discuss the future works.

2 System Implementation

Our lifelogger system consists of three major parts, namely, the Lifelogger Mobile Client, the Lifelogger Application Server and the Lifelogger Web Interface (`http://www.lifelogger.info`). Figure 2 depicts its overall software architecture.

2.1 Lifelogger Mobile Client

Thanks to the rapid advancement of commercial mobile devices, we are able to build our Lifelogger client devices directly from off-the-shelf products. Our LifeLogger Mobile Client is a helmet mounted with one Nokia N95 phone (Figure 1).

The software for the mobile client is written using the PyS60 SDK for the Symbian platform. The client records various types of sensory data including i)

accelerometer for motion, ii) GPS coordinates for outdoor locations, iii) camera view finder for pictures, iv) microphone recordings for sound, v) rotation sensor readings for rotation, and vi) WiFi signal strength for indoor locationing. These sensory data are all collected together with their corresponding timestamps. The data is captured in the JSON format and transmitted to the LifeLogger Application Server using the HTTP protocol via wireless connections.

Fig. 1. Sensor helmet for collecting activity data.

2.2 Lifelogger Application Server

The LifeLogger Application Server is responsible for storing, pre-processing, modeling recorded data as an activity language and finally retrieving similar activities of the user. It is also in charge of supplying the user interface for users to interact with their lifelogs.

The Server consists of mainly 4 components, namely - i)Indexing Service, ii) Similar Activities Retrieval Service, iii) Hierarchical Segmentation Service and iv) Activity Language Corpus Database. Once the mobile client transmits the sensor data to the server, the server pre-processes and converts the accelerometer readings into the activity language and stores the data to the datastore. The Indexing Service then indexes the activity language corpus. The Similar Activities Retrieval Service allows the user to select a portion of the lifelog and show similar activities performed by the user in the past. This feature allows the system to automatically label activities that are similar in nature to the one's labeled manually in the past by the user.

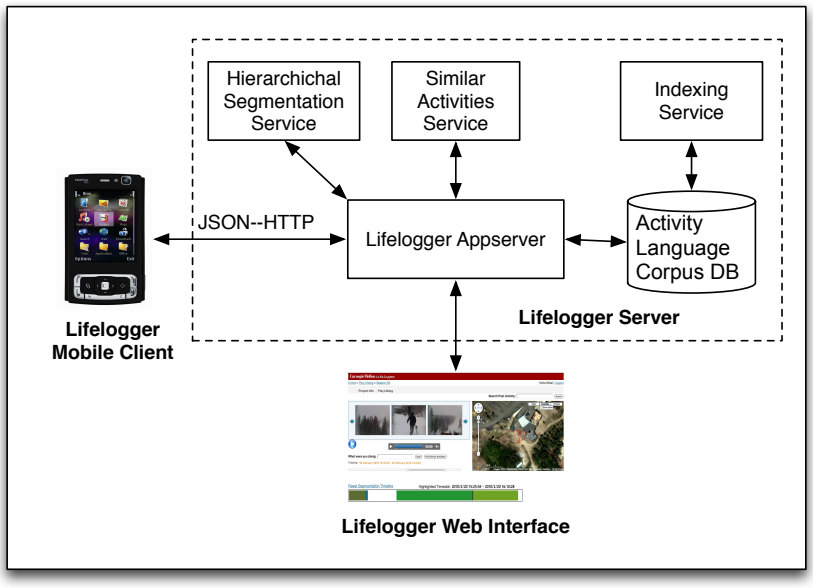

Fig. 2. System Architecture of the Mobile Lifelogger System

2.3 Lifelogger Web Interface

The server-side module is implemented as a web application so that users can access it easily through their favorite web browsers. The application provides an easy-to-use interface for end users to browse, annotate, and search their lifelogs. A personal calender based timeline is provided for all the recorded Lifelogs (Figure 3). The user can browse through the calendar and select a particular session that he's interested in revisiting his memory.

To make it easy for users to recall their past living experiences, we fit the collected images, audio, and GPS location data into one screen (Figure 4) to let users intuitively combine these memory clues. Moreover, since all sensory data are associated with synchronized time-stamps, users can navigate through the data set by simply dragging the timeline at the center of the screen, and all the three types of data would be updated simultaneously. The users are also provided with an option to "play" their lifelogs. This feature makes the images to automatically scrolls through, show the incremental trail of GPS on the map and play the audio at the same time. This acts as a memory cue so that the user can better recollect the experiences associated with that lifelog.

Users can also annotate a selected segment of lifelog by providing a short text description. Such text description will be used to learn the association between natural language query and the stored lifelog data. If the user is interested in a specific part of the lifelog and would like to find all his / her lifelog segments that

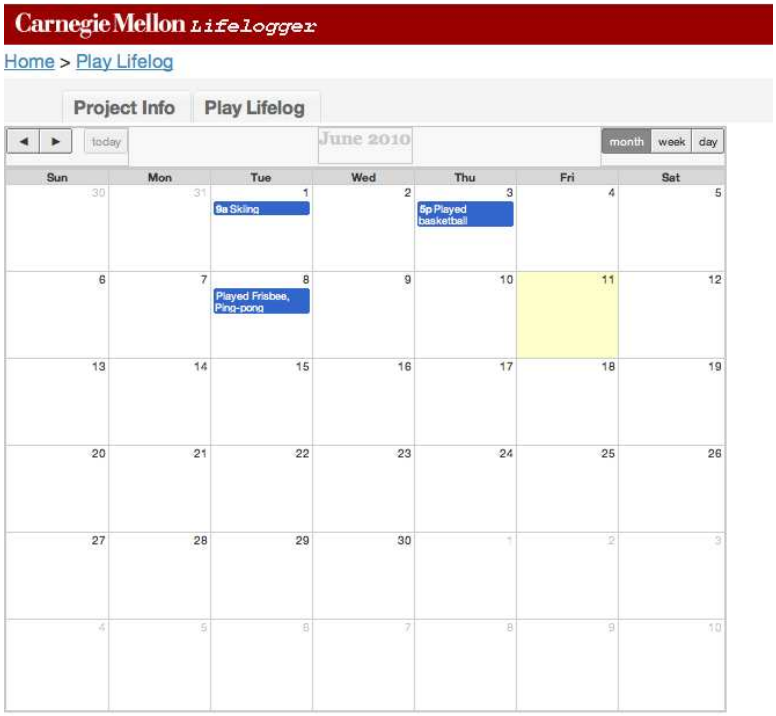

Fig. 3. Personal calendar based timeline

are similar to it, all the user need to do is to first select that specific lifelog part using the timeline control and then click the "Find Similar Activities" button beside it. The similar lifelog segments will be returned and listed in a table at the bottom of the screen (Figure 4) and are already sorted based on their relevance scores ranging from 0 to 1.

Users can also view hierarchical segmentation of different activities that they have performed. A color code is assigned to each activity that the user has performed over the period of the lifelog, so that he/she can easily distinguish between different activities. Similar activities have the same color code, making it easier for the user to identify what activities he has repeatedly performed.

3 Language-Based Indexing

3.1 Main Idea

One of challenges in indexing, retrieving and interpreting lifelogs is that a lifelog is a collection of heterogenous sensory information and each sensory data type requires a special method to process the raw input. In most existing lifelog applications, raw input from sensors is classified into predefined classes by trained

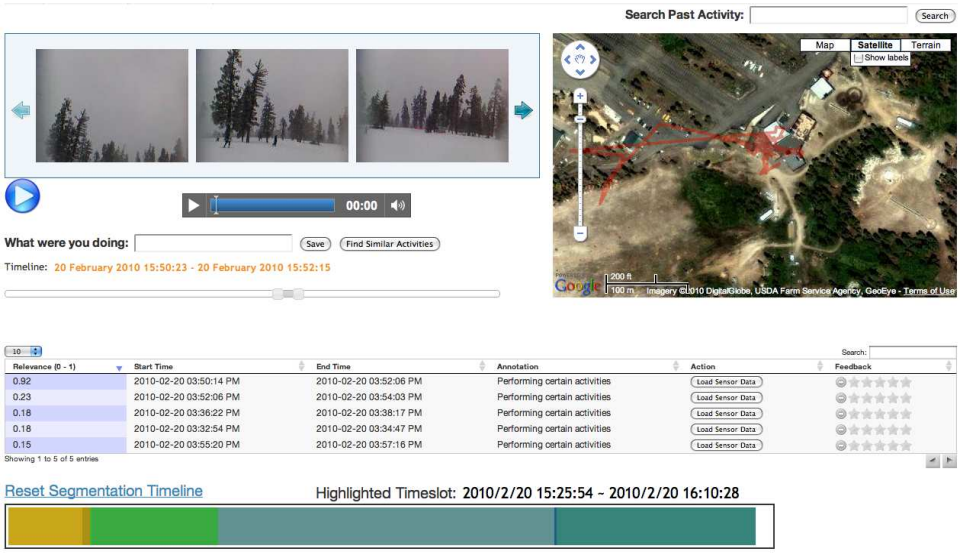

Fig. 4. Web interface of browsing/searching/annotating life logs.

classifiers for further processing. This usually limits the scope of lifelog applications to those predefined activities.

In this paper, we propose a novel method of representing the sensory input as " activity language" through quantizing the raw sensory input. Here we use *motion* information as an example. To record users' motion, we use 3-axis accelerometers to measures the acceleration at the X, Y, Z direction at the time of sampling. For the built-in accelerometers used in our experiments, the raw readings for each axis ranges from -360 to 360 which translates into 373,248,000 different (a_x, a_y, a_z) combinations. We quantize the raw accelerometer reading into V groups using K-Means clustering algorithm. Once the K-means clustering algorithm converges, it results in V cluster centroids and we give each cluster a label such as "D", "GC" and "DFR". We can then convert all the training and testing accelerometer data to their nearest cluster's label and thus convert the ambulatory activity into "activity text" (Figure 5).

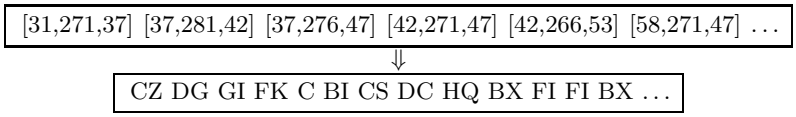

Fig. 5. An example of quantizing accelerometer readings to activity language representation.

There are three benefits from quantizing the raw sensory input into " activity language" representation:

- Dimension reduction of sensory input. High dimension input data is reduced into one dimension to reduce the computation complexity. In the case of accelerometer readings, the original 3-dimension input of (a_x, a_y, a_z) is now reduced to one dimension.
- Efficient indexing and searching of lifelogs. Searching an indexed text corpus is much easier than searching a database of real numbers. Compared to the infinite real number space, the limited "vocabulary" size of the text representation allows the search algorithm to be much more efficient. Index and search algorithms such as the Inverted Index and Suffix Arrays [11] developed for strings can be applied on the " activity language" representation of lifelogs. This is more straight forward than searching the lifelogs based on the cosine or Euclidean distances between the query and the logged activities.
- Uniformed representation of heterogenous sensory data. By converting different sensory input into the same type of " activity language" representation, we can develop and apply the same " activity language processing" algorithms on different types of data. Although the activity language discussed in this paper was constructed only from accelerometer readings, other types of sensory data can be used to generate activity languages as well. For example, in [5] we had demonstrated our preliminary results of using GPS recordings to generate activity languages.

We call this representation "activity language" based on the analogy between human activity and natural languages. The similarity between human activity and language had been articulated by Burke [4] and Wertsh [18]. Based on the "principle of language as action", natural languages and human activities indeed share some important properties. For instance, they are both "mediational means" or tools by which we achieve our ends. Additionally, they both exhibit structure and satisfy "grammars".

Natural Language	activity language	Example
Word	Atomic Movement	Turn upper body left
Phrase	Movement	Stand up
Sentence	Action	Climb up stairs
Paragraph	Activity	Enter building, climb up stairs and walk into office
Document	Event	Left home and ride bicycle to campus arrived at my office at 2nd floor

Table 1. Activity as language at different levels.

Table 1 illustrates that people's ambulatory activities share a lot in common with natural languages at all levels. The anatomy of human bodies allows us to perform certain *atomic movements* such as "turn upper body left" whereas

"jump up at 10g acceleration" is not possible. Such atomic movements form the vocabulary of the activity language. A sequence of atomic movements performed in a meaningful order creates a *movement* such as an *action* of "standing up". *Actions* such as "climbing up stairs" are created by performing actions in a right order similar to create a "sentence". A sequence of actions builds up an *activity*. The higher level concept *event* is composed of a series of activities in a similar way as a *document*.

This " activity language" concept serves as the foundation for our approach to efficient lifelog retrieval. To empirically evaluate the similarity between the ambulatory activity and natural languages, we check if the activity language corpus follows the Zipf's law. Zipf's law states that given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table. In other words, the logarithm of a word's frequency is linear to its rank in a natural language corpus. Figure 6 plots the logarithm frequency of word types in the activity language corpus for word type ranked 1, 2, 3, etc. Though not exactly linear, it does plot a line similar to Zipf's distribution.

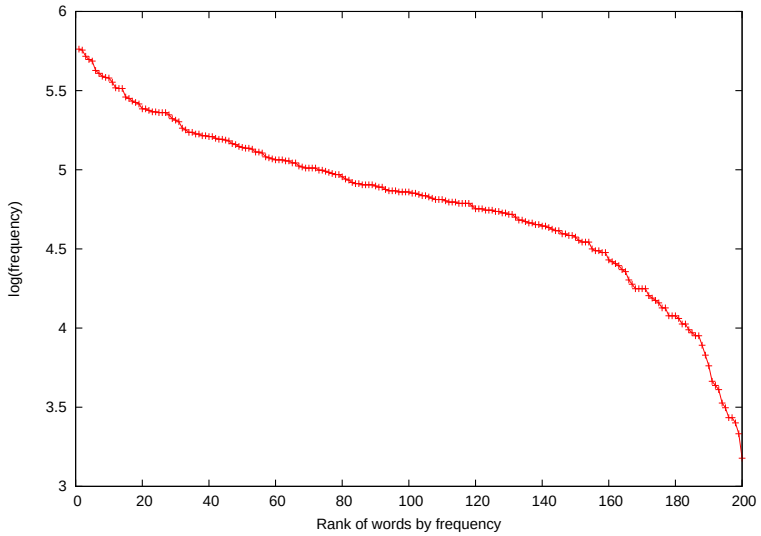

Fig. 6. log(freq) vs. rank of frequency of word types in a lifelog converted as activity language.

3.2 Activity Recognition

Before applying the "activity language" concept in our lifelog system, we performed a series of predefined-activity recognition experiments to justify the benefits of modeling human activities as a language.

In our approach, we view the labeled data for each activity a_i as the training corpus and train a smoothed n-gram language model over the converted activity language text using the SRI language model toolkit [16]. For each testing "activity sentence" t, we input it to all the pre-built language models to calculate the probability of t being generated by activity a_i and predicts the activity of the testing sentence to be i^* such that

$$i^* = \arg\max_i P(t|a_i) \tag{1}$$

One issue of using language models for activity recognition is that language model probabilities are not directly comparable if their respective training data have different vocabulary sizes. To solve this problem, each training data set is augmented with a universal vocabulary list built from all training data sets. As a result, all our activity language models have the same vocabulary size, and thus their generated probabilities are comparable.

	Predicted Activity		
	walking	running	cycling
walking	94%	3%	3%
running	6%	92%	2%
cycling	8%	0%	92%

Table 2. Classification accuracy on corpus with vocabulary=100.

	Predicted Activity		
	walking	running	cycling
walking	95%	1%	4%
running	4%	94%	2%
cycling	2%	0%	98%

Table 3. Classification accuracy on corpus with vocabulary=200.

Our preliminary results of using smoothed n-gram language model for activity recognition demonstrated an average accuracy rate of 94% in distinguishing among basic activities such as walking, running, and cycling. Table 2 and 3 compare the recognition accuracy of language models trained over a corpus of vocabulary size 100 vs. the one with 200 word types. With a larger vocabulary size, i.e., more atomic movement types, the activity language has more discriminative power to differentiate human activities.

Figure 7 shows the average activity recognition accuracy vs. the order of n in language model training. Overall, for this basic activity recognition task, the order of history does not play a significant role here.

The promising results of these experiments increase our confidence in using the activity language to improve lifelog systems' indexability.

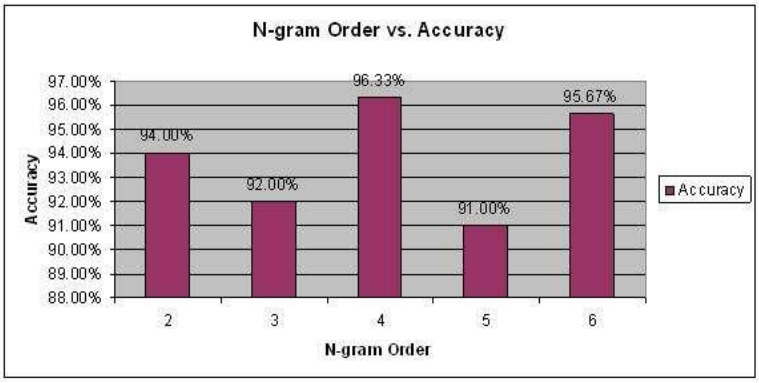

Fig. 7. Recognition accuracy vs. n-gram order.

3.3 Similar Activity Retrieval

In many cases, we want to find out information about activities that are not predefined such as "how many tennis games did I play in the past two months?" or "how much time did I spend sitting in front of TV last week?" It is not possible to enumerate all possible activities and train Hidden Markov Models ahead of time in order to answer such questions. In our approach, we convert the lifelogs, in particular, the main indexing sensory information into a text representation. This allows us to apply Information Retrieval techniques to "retrieve' relevant activities from the past logs to answer users' queries.

In our implementation, a user can selects a segment from his/her lifelogs on the web interface and indicate that he/she may want to find similar activities from the past logs. The highlighted segment does not need to be annotated by natural language descriptions such as "playing tennis". The system will search from the past life logs and return most relevant segments for user to review.

The key here is to calculate "similarity" between two segments of lifelogs. Inspired by the BLEU metric [14] where averaged n-gram precision is used to measure the similarity between a machine translation hypothesis and human generated reference translations, we use *averaged n-gram precision* to estimate the similarity between two lifelog segments.

Assuming that P and Q are two activity language sentences of the same length l. P is the sequence of P_1, P_2, ..., P_L and Q is the sequence of Q_1, Q_2, ..., Q_L. Denote the *similarity* between P and Q as $S(P,Q)$. Define the n-gram precision between P and Q as $\text{Prec}_n(P,Q) =$

$$\frac{\sum_{\tilde{p}\in\{\text{All } n\text{-gram types in } P\}} \min(freq(\tilde{p},P), freq(\tilde{p},Q))}{\sum_{\tilde{p}\in\{\text{All } n\text{-gram types in } P\}} freq(\tilde{p},P)}, \tag{2}$$

and the similarity between P and Q is defined as:

$$S(P,Q) = \frac{1}{N}\sum_{n=1}^{N} \text{Prec}_n(P,Q) \quad (3)$$

$\text{Prec}_n(P,Q)$ calculates the percentage of n-grams in P that can also be found in Q and $S(P,Q)$ averages the precision over 1-gram, 2-gram and up to N-gram. In our experiments, we empirically set $N = 5$.

Table 4 shows an example of calculating the similarity between activity sentence P ("NB NB P P P P P P NB NB") and Q ("NB P P NB NB P NB P P P").

n	n-gram	$freq_P$	$freq_Q$	min	Prec_n
1	NB	4	4	4	
	P	6	6	6	
					10/10=1.0
2	NB NB	2	1	1	
	NB P	1	3	1	
	P P	5	3	3	
	P NB	1	2	1	
					6/9 = 0.67
3	NB NB P	1	1	1	
	NB P P	1	2	1	
	P NB NB	1	1	1	
	P P NB	1	1	1	
	P P P	4	1	1	
					5/8 = 0.63
4	NB NB P P	1	0	0	
	NB P P P	1	1	1	
	P P NB NB	1	1	1	
	P P P NB	1	0	0	
	P P P P	3	0	0	
					2/7 = 0.29
5	NB NB P P P	1	0	0	
	NB P P P P	1	0	0	
	P P P NB NB	1	0	0	
	P P P P NB	1	0	0	
	P P P P P	2	0	0	
					0/6 = 0.0
					$S(P,Q) = 0.52$

Table 4. Calculating the similarity between two activity sentences using averaged n-gram precision.

Given a query sentence of l words, we assume that similar activities in the lifelog should also be of length l. This assumption makes the retrieval algorithm easier to implement as varied length activity retrieval would require activity segmentation. For a lifelog with G words, there are $G - l$ different strings of l words long. In our current setting, a 24 hours lifelog contains about 200 million

activity words. Calculating the similarity between each of the $G-l$ strings with the query can be computationally expensive. To speed up the retrieval, we use suffix arrays to pre-select strings in the corpus that have high order n-gram matches with the query and calculate $S(P,Q)$ scores for those strings only. The observation is that if a string in the lifelog is similar to the query, then it should have many high order n-grams matched with those in the query string.

Top R similar activity segments is returned to the user on the web interface (as shown in lower panel in Figure 4). User can load each segment to "play" the corresponding lifelogs and for our ongoing experiments evaluate if the segment is truly "similar" to the query.

3.4 Hierarchical Segmentation of Lifelogs

Lifelog records a user's daily life as a continuous sequence of sensory data. After converting the sensory data to activity language text, a lifelog is now a long string of text. Just as we need punctuations, sentence boundaries and paragraph boundaries in written text, it would make lifelogs more readable if we could automatically segment the data based on user's activities.

Configuration	Value
Activity Type	Playing frisbee, Playing basketball
	Playing table tennis, Playing tennis
Activity Duration	5 to 10 minutes each
Lifelog Length	40 to 50 minutes each
Accuracy Measure	F1 score

Table 5. Configuration of the automatic segmentation experiments.

The underlining assumption of our segmentation algorithm is that when a user switches his/her activity at time t, the similarity between string $[t-w, t-1]$ and $[t, t+w]$ should be much lower than if t is inside the same activity for a window of size w. For a window size w, define the "change of activity" at time t as:

$$H(t,w) = -log(0.00001 + S([t-w, t-1], [t, t+w-1])). \quad (4)$$

The higher the value of $H(t,w)$, the more likely user changed his/her activity at time t. Figure 8 shows the H value at each data points given different window sizes for a segment of lifelog.

It can be noticed that: (1) peaks of activity change identified by larger windows are also peaks identified by smaller windows but not vice versa; and (2)activity changes over larger windows are smoother than smaller windows. Intuitively, larger window size captures changes of larger-scale activities whereas smaller window captures changes of smaller activities. Based on this finding, we first segment the lifelog data using large window size and then recursively segment the data using smaller windows. This results in a hierarchical segmentation

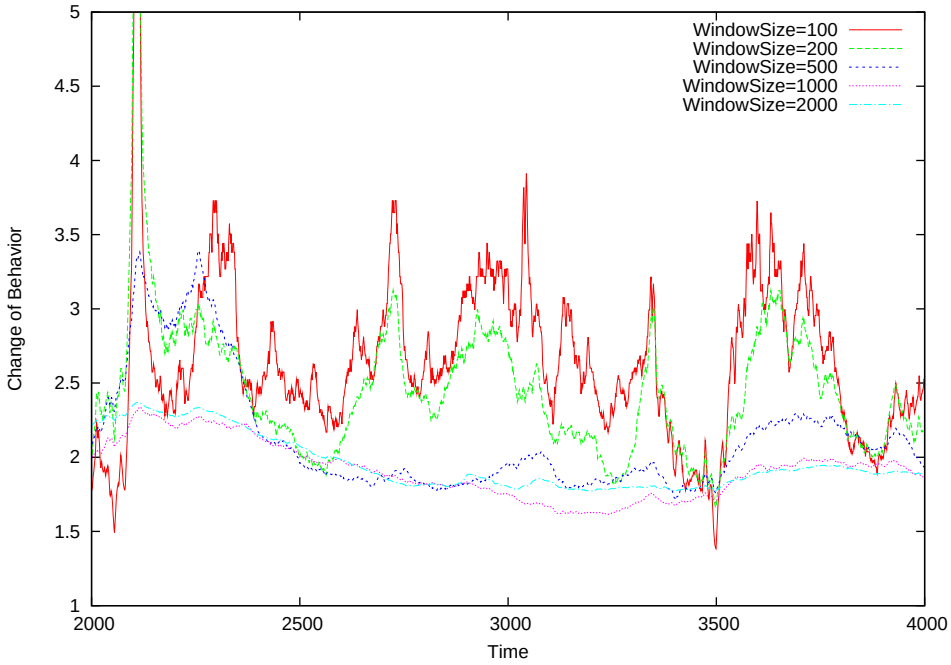

Fig. 8. Activity changes calculated by different size of sliding windows.

of lifelogs which allows user to efficiently browse through the lifelog instead of playing the whole lifelog (Figure 10).

To evaluate the quality of this automatic segmentation method, experiments were performed with the configuration shown in Table 5. For every experiment, the participant would wear the sensor helmet to perform a series of different activities and there would be an observer who was in charge of recording the timestamps at which the activity changes occur. The collected human annotations would then serve as ground truth to help us evaluate the accuracy of the automatic segmentation. Only the first level of the segmentation would be evaluated and the metric we used is the F1 score calculated as

$$F = 2 * precision * recall/(precision + recall). \quad (5)$$

Table 6 shows the F1 scores of our automatic segmentation method according to different window sizes. Overall, our segmentation method achieved an F1 score of 60%.

3.5 Segmentation Clustering of Lifelogs

Hierarchical segmentation of the lifelogs shows different activities performed by the user over time. However, it would be further useful if the system groups similar segments so that user perceives similar activities he/she has performed.

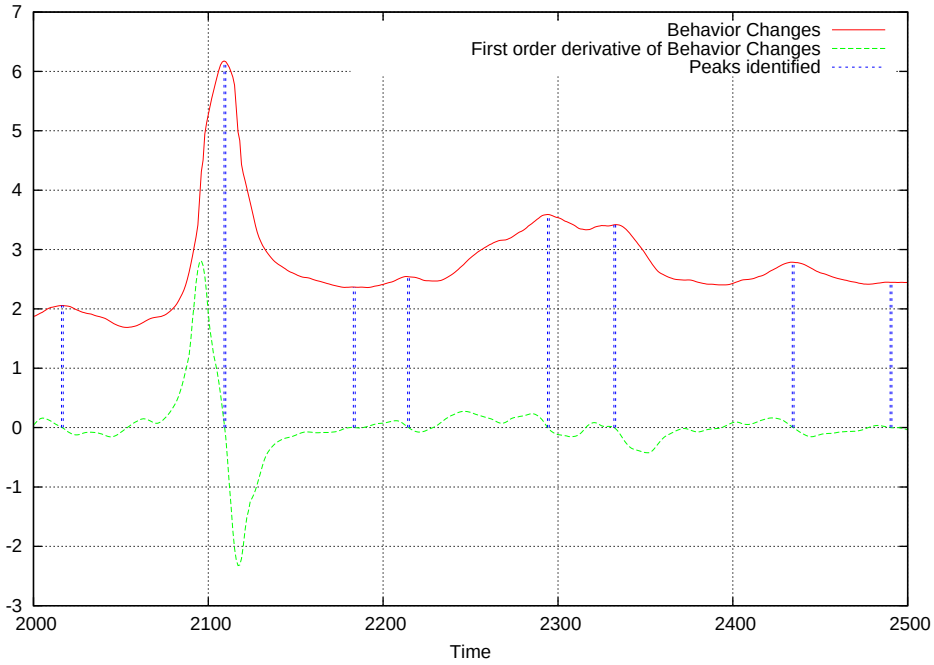

Fig. 9. Identifying peaks in the behavior-change-curve.

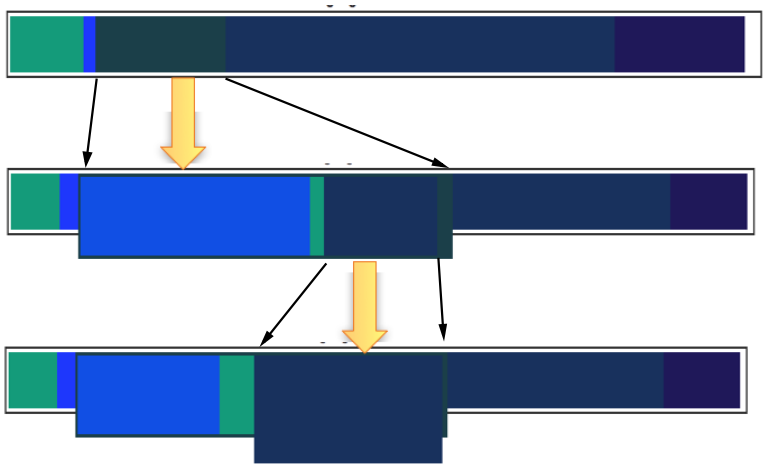

Fig. 10. Hierarchically segmented lifelog. Activity boundaries detected automatically by the system and descriptions are added by the user.

By performing unsupervised clustering on the Autosegmentation ouput, we can group similar activities.

K-means clustering algorithm is chosen for its simplicity and performance [7]. As it requires similarity or distance between two segments, we use a similarity measure analogous to Section 3.3 - *averaged n-gram precision*. But, the key difference is that the segments need not be of the same length.

Let P and Q be two activity language sentences, with lengths l and m respectively. P is the sequence of P_1, P_2, ..., P_L and Q is the sequence of Q_1, Q_2, ..., Q_M. Denote the *similarity* between P and Q as $S(P,Q)$. Define the n-gram precision between P and Q as $\text{Prec}_n(P,Q) =$

$$\frac{1}{2} \sum_{\tilde{p} \in \{\text{All } n\text{-gram types in } P \text{ or } Q\}} \min(freq(\tilde{p},P), freq(\tilde{p},Q)) \left(\frac{1}{\sum_{\tilde{p} \in \{\text{All } n\text{-gram types in } P\}} freq(\tilde{p},P)} + \frac{1}{\sum_{\tilde{q} \in \{\text{All } n\text{-gram types in } Q\}} freq(\tilde{q},Q)} \right) \quad (6)$$

and the similarity between P and Q is defined as:

$$S(P,Q) = \frac{1}{N} \sum_{n=1}^{N} \text{Prec}_n(P,Q) \quad (7)$$

Each sentence is vectorized using the frequencies of all its n-grams. Each dimension of the vector represents one n-gram type. Using the same example in Section 3.3, an activity sentence P ("NB NB P P P P P P NB NB") and Q ("NB P P NB NB P NB P P P"). Therefore, the vectors for sentence P and Q are constructed as

$$V_P = (4,6,2,1,5,1,1,1,1,1,4,1,1,1,1,3,1,1,1,1,2)$$
$$V_Q = (4,6,1,3,3,2,1,2,1,1,1,0,1,1,0,0,0,0,0,0,0)$$

Each component of the these vectors is the frequency of one of the n-gram types in the n-gram column of Table 4. Note that if a particular n-gram does not appear in a sentence, the component of that n-gram is set to zero. e.q. n-gram ("NB NB P P P") in sentence Q.

The centroid of cluster is constructed by calculating the ***average vector*** of all the vectors in the given cluster.

We have applied this method to the output of hierarchical segmentation to generate clustering results at different levels. The preliminary results show that the k-means algorithm can converge quickly, within 3-4 iterations, when running with a small number of clusters as 5. Due to the limited volume of annotated data, we could not evaluate the results in a systemical manner. However, manual cross-checking between the clustering results and the user annotations at the top level shows some level of relevance. This is outside of the scope of this paper and we will leave it for future work.

Window Size	100	200	300	400	500	600	700	800	900	1000
Precision (%)	26.09	40.00	43.75	46.67	41.67	44.44	44.44	50.00	50.00	66.67
Recall (%)	60.00	80.00	70.00	70.00	50.00	40.00	40.00	50.00	40.00	60.00
F1 Score (%)	36.36	53.33	53.84	56.00	45.45	42.11	42.11	50.00	44.44	63.16

Table 6. F1 Score vs. Window Size

4 Related Work

4.1 Activity Recognition

There have been several techniques for recognizing or distinguishing basic human activities. They can be categorized into two flavors: heuristic threshold-based classifiers and pattern recognition techniques such as decision trees, nearest neighbor, Naive Bayes, support vector machines (SVM), neural networks, Hidden Markov Models (HMM) and Gaussian mixture models [13]. For recognizing high-level human activities, several attempts had been made in [1, 15]. Among these techniques, the most popular ones we see so far are those based on HMM. These HMM-based approaches classify the input sensory information into one of the predefined activities such as walking, running, and standing. However, since HMM assumes the first order Markov chain in the state space and usually does not consider the inherent "grammar" or "structure" of human activities, the activities that can be recognized by HMM are limited to those pre-defined in the training data, which as a result limits HMM's application in people-centric computing.

4.2 Lifelog System

Different approaches have been used to implement a lifelog system. The MyLifeBits system [6] is designed to store and manage everything in a person's lifetime that can be captured in digital format. Its initial goal was to store all personal information found in PCs such as articles, video, office documents, email, keystrokes, and screen mouse clicks, etc. It then evolved into storing all ambient information of a person's daily life via a specialized camera device named SenseCam. MyLifeBits supports capture, storage, management, and retrieval of many media types, and its sophisticated database design is capable of storing a large volume of multimedia data. However, MyLifeBits only applies a basic metadata-based indexing approach which requires users to manually annotate most of the collected data in order to have meaningful search results. Our works address this issue well by providing a more effective indexing scheme which requires less user involvement and provides more meaningful search results by taking the "meaning" of the collected media into account. Another lifelog system implementation is discussed in [8]. This work focuses on realtime storage and retrieval of lifelog in a ubiquitous environment. The developed system supports semi-automatic activity analysis and provides an intuitive graphical interface for users to browse

their lifelogs that correlates the space and temporal information of the displayed sensory data.

In addition to our language approach to indexing lifelog, Kim et al presented a multimodal sensor fusion technique which supports automatic generation of lifeline's metadata [9]. The key idea is to combine the analysis results of different kinds of low-level sensory data to better infer higher-level context information about the collected lifelog. For example, by combining the analysis of audio, GPS, and accelerometer readings, the system is able to better identify the environment in which the lifelog was taken. Machine learning techniques such as decision tree and Gaussian Mixture Model (GMM) are used to analyze the collected low-level sensory data. Similar techniques to this sensor fusion approach are explored in [2, 17]. The former uses video key frame summarization and conversation scene detection to fulfill efficient lifelog retrieval. The latter proposes an integrated technique to process lifelog data using correlations between different types of the captured data from multiple sensors.

Kyoko et al. [19] have developed a wearable lifelogging device to recognize the experiences and activities of cats and post them as tweets on Twitter. C4.5 decision tree is used to recognize and classify different activities performed by cats such as eating, sleeping running.

5 Conclusions and Future Work

In this paper, we present our Mobile Lifelogging system to record, index and understand the life experiences of a mobile user. We discuss several functions of the system such Similar Activity Retrieval (Section 3.3) and Automatic Hierarchical Segmentation (Section 3.5) for efficiently retrieving the lifelogs and help the user visualize them. We present a novel yet straightforward approach of processing lifelogs of heterogenous sensory data. We verify the similarity between activity and language by demonstrating Zipf's distribution over our activity language corpus. The experimental results presented in Section 3.2 demonstrate high accuracy of using language models for human activity recognition. Unlike the traditional HMM approach which is limited to activity recognition task, modeling activity as language enables many other applications such as similar activity retrieval, and hierarchical activity segmentation. Besides, this language-based modeling approach can be applied to other types of sensory input such as geolocations. Extracting high level semantic information from primitive activities that are recognized from the sensory data still remains a challenging problem. We will conduct user study to evaluate the effectiveness of our similar activity retrieval service. We will also carry out more experiments to generate sufficient annotated data. With these data, we hope to extend our work of hierarchical activity segmentation by grouping atomic sensory events in the sequence progressively to build abstraction hierarchy of HHMM towards automatic lifelog summarization.

References

1. R. Aipperspach, E. Cohen, and J. Canny. Modeling human behavior from simple sensors in the home. In *Proceedings of IEEE Conf. on Pervasive Computing*, pages 337–348, Dublin, Ireland, April 2006.
2. K. Aizawa, D. Tancharoen, S. Kawasaki, and T. Yamasaki. Efficient retrieval of life log based on context and content. In *CARPE'04: Proceedings of the the 1st ACM workshop on Continuous archival and retrieval of personal experiences*, pages 22–31, New York, NY, USA, 2004. ACM.
3. R. Brookmeyer, E. Johnson, K. Ziegler-Graham, and H. M. Arrighi. Forecasting the global burden of alzheimer's disease. *Alzheimer's and Dementia*, 3(3):186–191, July 2007. predicted 107 million people will suffer from Alzheimer by 2050.
4. K. Burke. *Language as Symbolic Action*. University of California Press, 1966.
5. P. Chen, S. Chennuru, S. Buthpitiya, and Y. Zhang. A language-based approach to indexing heterogeneous multimedia lifelog. In *Proceedings of 12th International Conference on Multimodal Interfaces*, 2010.
6. J. Gemmell, G. Bell, R. Lueder, S. Drucker, and C. Wong. Mylifebits: fulfilling the memex vision. In *MULTIMEDIA '02: Proceedings of the tenth ACM international conference on Multimedia*, pages 235–238, New York, NY, USA, 2002. ACM.
7. J. A. Hartigan. *Clustering Algorithms*. ISBN 0-471-35645-X. Wiley, 1975.
8. I. jae Kim, S. C. Ahn, and H. gon Kim. Personalized life log media system in ubiquitous environment. *Lecture Notes in Computer Science*, 4412, 2006.
9. I.-J. Kim, S. C. Ahn, H. Ko, and H. G. Kim. Automatic lifelog media annotation based on heterogeneous sensor fusion. In *Proceedings of IEEE International Conference on Multi Sensor fu-sion and Integration for Intelligent systems*, Seoul, Korea, August 20-22 2008.
10. M. L. Lee and A. K. Dey. Lifelogging memory appliance for people with episodic memory impairment. In *UbiComp '08: Proceedings of the 10th international conference on Ubiquitous computing*, pages 44–53, New York, NY, USA, 2008. ACM.
11. U. Manber and G. Myers. Suffix arrays: a new method for on-line string searches. *SIAM J. Comput.*, 22(5):935–948, 1993.
12. L. Neergaard. Report: 35 million-plus worldwide have dementia. *Associate Press*, Sep. 21 2009.
13. A. Nguyen, D. Moore, and I. McCowan. Unsupervised clustering of free-living human activities using ambulatory accelerometry. In *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, pages 4895–4898, Lyon, France, Aug 22-26 2007.
14. K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. Technical Report RC22176(W0109-022), IBM Research Division, Thomas J. Watson Research Center, 2001.
15. D. Patterson, L. Liao, D. Fox, and H. Kautz. Inferring high-level behavior from low-level sensors. In *Proceedings of the Fifth International Conference on Ubiquitous Computing (UbiComp 2003)*, pages 73–89, Seattle, Washington, October 12-15 2003.
16. A. Stolcke. Srilm – an extensible language modeling toolkit. In *Proc. Intl. Conf. on Spoken Language Processing*, volume 2, pages 901–904, Denver, CO, 2002.
17. K. Takata, J. Ma, B. O. Apduhan, R. Huang, and Q. Jin. Modeling and analyzing individual's daily activities using lifelog. In *ICESS '08: Proceedings of the 2008 International Conference on Embedded Software and Systems*, pages 503–510, Washington, DC, USA, 2008. IEEE Computer Society.

18. J. V. Wertsch. *Mind As Action*. Oxford University Press, USA, 1998.
19. K. Yonezawa, T. Miyaki, and J. Rekimoto. Cat@log: sensing device attachable to pet cats for supporting human-pet interaction. In *ACE '09: Proceedings of the International Conference on Advances in Computer Enterntainment Technology*. ACM, 2009.