LightSeg: An Online and Low-Latency Activity Segmentation Method for Wi-Fi Sensing

Liming Chen, Xiaolong Zheng(✉), Leiyang Xu, Liang Liu, and Huadong Ma

School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China
{chenliming1997,zhengxiaolong,xuleiyang,liangliu,mhd}@bupt.edu.cn

Abstract. WiFi based activity recognition mainly uses the changes of Channel State Information (CSI) to capture motion occurrence. Extracting correct segments that correspond to activities from CSI series is then a prerequisite for activity recognition. Researchers have designed various segmentation methods, including threshold-based and deep learning-based methods. However, threshold-based methods are highly empirical and the threshold is usually dependent on the application and environment. When dealing with mixed-grained activities, the predefined threshold will fail. On the other hand, deep learning-based methods are impractical for online systems with low-latency demand because of their high overhead. In this paper, we propose LightSeg, an online and low-latency segmentation method leveraging an activity granularity-aware threshold that quickly adjusts itself based on the granularity of the activity in the current detecting window. We propose a threshold post-decision mechanism that detects the end of a segment first and then decides the appropriate threshold based on the most recent activity. By this way, LightSeg automatically adapts to different activity granularity in practice. Compared to existing threshold-based methods, LightSeg greatly reduces the dependence on expertise to decide the threshold. Experimental results show that LightSeg improves the segmentation accuracy by up to 14% compared to the existing threshold-based method and reduces the data processing time by 97% compared to the deep learning-based method.

Keywords: WiFi sensing · CSI · Activity segmentation

1 Introduction

In recent years, wireless sensing has attracted more and more attention from both academy and industry. It leverages wireless signals such as WiFi, Bluetooth, RFID, and ZigBee, to sense the behaviors of a person or the states of

This work is supported in part by the National Natural Science Foundation of China (No. 61932013), the A3 Foresight Program of NSFC (No. 62061146002), and the Funds for Creative Research Groups of China (No. 61921003).

S. Longfei and P. Bodhi (Eds.): MobiQuitous 2022, LNICST 492, pp. 227–247, 2023.
https://doi.org/10.1007/978-3-031-34776-4_13

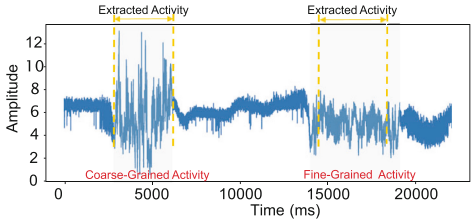
other objects, and then realize smart applications [17,19,24–26]. Since WiFi is ubiquitous and not affected by illumination, WiFi-based wireless sensing has been widely studied.

Since WiFi Channel State Information (CSI) contains finer-grained and more stable channel information [14] and can be extracted from commercial WiFi devices [20], it is commonly used in existing WiFi sensing applications. The complex-valued CSI contains amplitude and phase information. CSI phase is often interfered by uncertain disturbances such as carrier frequency offset (CFO) and sampling frequency offset (SFO) [21]. While the amplitude of CSI is the generally reliable and accessible metric for activity extraction and classification [21], thus CSI amplitude is a commonly used feature for WiFi sensing.

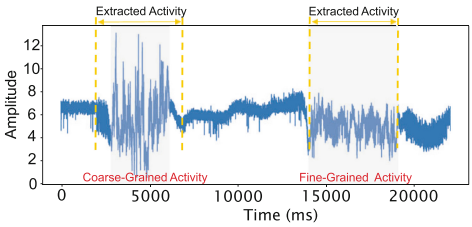
After obtaining CSI, two major stages are needed for WiFi sensing: activity segmentation and activity classification [2]. Accurate activity segmentation is a prerequisite of accurate activity classification. The most common activity segmentation method used in WiFi sensing is the threshold-based method. Existing works uses a predefined fixed threshold to extract activity [1,7,8,10,15]. These fixed-threshold methods have to set empirical parameters and select an appropriate denoising strategy to determine a good threshold. Some works take the environmental factors such as noise into account when determining the threshold [9,11,18,23]. They adjust the threshold automatically according to the noise level in the environment.

Most of the existing threshold-based segmentation methods only adjust the threshold based on noise but ignore the granularity of target activity. Different activities with different granularity can desire different and even contradictory thresholds. Fine-grained activities such as gestures require a small threshold in case missing the meaningful small CSI changes, but coarse-grained activities such as postures (falling, gait, etc.) desire a large threshold to exclude more noise. In practice, coarse-grained and fine-grained activities usually occur alternately and randomly. Then the threshold decided based on one granularity will extract incorrect CSI segments.

Take the application of elderly monitoring as an example. A target may walk to the living room, take a drink, and then sit down on a chair. The activity recognition system needs to continuously segment the coarse-grained activities (walking and sitting down) and the fine-grained activity (drinking). For existing methods, the predetermined threshold for activities of one granularity is not appropriate for the other, resulting in the decline of segmentation accuracy and affecting the accuracy of activity recognition. For example, for a pair of coarse-grained and fine-grained activities shown in Fig. 1, since the CSI changes caused by the coarse-grained activity are distinguishable from background CSI noise, a large threshold is appropriate, which can avoid the extracted activity containing unnecessary noise. But applying the large threshold to the fine-grained activities, meaningful CSI parts can be misjudged as noise and the extracted CSI segmentation can be incomplete, as shown in Fig. 1(a). On the contrary, if a small threshold appropriate for the fine-grained activity is applied to segment the coarse-grained activity, some unnecessary background noise might be included after activity extraction, as shown in Fig. 1(b).



(a) Use threshold appropriate for the coarse-grained activity.



(b) Use threshold appropriate for the fine-grained activity.

Fig. 1. Illustration of inaccurate segmentation of existing threshold-based methods for mixed-grained activities. The gray parts are groundtruth. (Color figure online)

Existing threshold-based methods only set a threshold in an activity detection window according to activities with one granularity for a particular application. But before activity extraction, it is difficult to set an appropriate threshold for all kinds of activities without knowing their granularity, especially when there are multiple activities in one detection window. The lack of awareness of activity granularity is the fundamental reason why the existing threshold-based segmentation methods are not suitable for activities with mixed granularity.

A deep learning-based segmentation method [16] is proposed to segment activities with arbitrary granularity. Because the deep learning model can learn the activity granularity through training, the work has achieved remarkable results. However, the overhead (memory, latency, hardware requirements) of deep learning-based methods is much higher than threshold-based methods and even unaffordable to embedded WiFi devices. In practice, online activity recognition demands low latency but the deep learning-based segmentation will be time-consuming and hard to provide friendly WiFi sensing services.

A practical activity segmentation method for WiFi sensing should be able to segment multiple activities with different granularity in a detection window with low latency. To obtain such a method, several requirements should be satisfied: (1) low overhead and fast granularity perception, (2) environment independence, and (3) interference resistance. By satisfying these requirements, we propose LightSeg, a lightweight granularity-aware threshold-based activity segmentation method that can segment activities with different granularity with low latency to support online WiFi sensing applications. We adopt fast preprocessing policies and remove the redundant information of the CSI data first. Since the variation of CSI amplitude reflects the intensity of activities, then we use the moving

variance in preprocessing module to obtain the one-dimensional preprocessed CSI series for fast granularity perception. Then we design a core segmentation algorithm with time complexity of only $O(N)$ that contains granularity-aware threshold decision and abnormal peak removal. Based on the estimated granularity, LightSeg decides the threshold appropriate to the target activity in the current detection window. We first use this threshold to determine the end point of an activity and then look up the corresponding start point. By this way, LightSeg avoids using a predefined threshold and can adjust the threshold for each activity regardless of the environment influence. To avoid incorrect segmentation caused by interference, we propose an abnormal peak removal mechanism that uses a valid duration to distinguish activities and interference.

In summary, the main contributions are as follows:

- We propose an online and low-latency activity segmentation method for WiFi sensing. We design an activity granularity-aware threshold adjusting algorithm, which can solve the performance degradation of activity segmentation with mixed or unknown granularity.
- We solve several practical issues of LightSeg. We design the granularity perception method, granularity-based threshold adjustment, and abnormal peak removal to improve the segmentation performance in practical scenarios.
- Experimental results show that the segmentation accuracy of LightSeg is up to 14% higher than the state-of-the-art threshold-based method. Compared to the deep-learning based method, LightSeg achieves similar and even better accuracy but has much less overhead in terms of both computational latency and resources.

2 Related Work

Activity segmentation is a prerequisite of classification. Many works have been carried out on activity segmentation [6,13]. These segmentation methods can be categorized as threshold-based methods and deep learning-based methods.

2.1 Threshold-Based Activity Segmentation

Threshold-based activity segmentation methods use a threshold to identify the start and end points and extract the attached CSI segment as valid activity data [5], because the variation of CSI amplitude is detectable with the onset of activities. A fixed threshold is easy to implement but its performance depends on applications and environments. Virmani and Shahzad [8] found that principal component analysis (PCA) can effectively remove the noise in CSI streams. And then they use a fixed threshold on the second principal component of the CSI stream to identify the occurrence of activities. Wu et al. [15] proved that the general PCA could not meet scenarios of the signals through a wall. Hence, they introduced the opposite robust PCA and designed a normalized variance sliding window algorithm to process the first principal component for activity segmentation. Sheng et al. [7] presented a segmentation algorithm based on

time series difference. The algorithm uses the mean absolute differences and the pre-set start threshold and end threshold to detect activities. Wang et al. [10] presented a two-stage segmentation algorithm to identify falling, they use a threshold-based sliding window method to identify whether the CSI waveforms are in a fluctuating state or a stable state. Then a backtracking window from the end point is used to detect the start point of a fall. Bu et al. [1] recognized the start and end of activities by calculating whether the variance in the window exceeded the empirical threshold, and presented a motion pause buffer algorithm to solve the wrong gesture segmentation caused by some short pauses in the gesture process.

To cope with dynamic noise, some segmentation methods adopt dynamic threshold. Feng et al. [3] divided the CSI stream in a detection window into several slots. Then the average difference between the sum of the largest half values and the sum of the smallest half values of CSI variance in each time slot is regarded as the threshold. Wang et al. [9] calculated the activity amplitude threshold based on the active and static CSI streams and updated the threshold based on newly collected CSI data to adapt to environmental changes. Yan et al. [18] presented an adaptive activity segmentation algorithm that updates the threshold based on CSI changes in the sliding window to balance the trade-off between performance and robustness. Wang et al. [11] used an Exponential Moving Average algorithm to update the noise level estimation and set the threshold three times as large as the noise level. Zhang et al. [23] presented kurtosis as the metric to evaluate power spectral density distribution (PSD). The detection threshold is set as three times as large as the PSD kurtosis when there is no movement. And an Exponential Moving Average algorithm is leveraged to update PSD kurtosis when there is no movement.

Both threshold-based segmentation studies focus on either fine-grained activities or coarse-grained activities and only adjust the threshold according to noise without consideration of the granularity of the target activity. Due to the unawareness of granularity of current activities in practice, the predefined threshold for specific activities can be inappropriate for other activities.

2.2 Deep Learning-Based Activity Segmentation

With the development of deep learning technique, researchers are applying deep learning to activity segmentation for WiFi sensing. Xiao et al. [16] proposed DeepSeg, a deep learning-based activity segmentation method. By using a large training dataset containing various activities with different granularity, the trained activity segmentation model can deal with the granularity unawareness problem.

However, the deep learning-based method is labor-intensive because it requires collecting a large amount of data with manually labeled ground-truth start and end points for each collected activity trace. What's worse, it requires huge computation and memory resources, which is hard to be affordable for embedded WiFi devices. Even if possible, the massive calculation of a deep learning model will be extremely time-consuming and lead to huge delay, hindering practical usages of online WiFi sensing application systems.

3 Design of LightSeg

In this section, we first present an overview of LightSeg and then introduce the design details of each module.

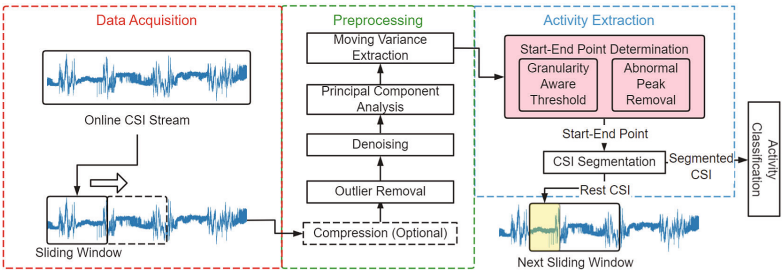


Fig. 2. Overview of LightSeg.

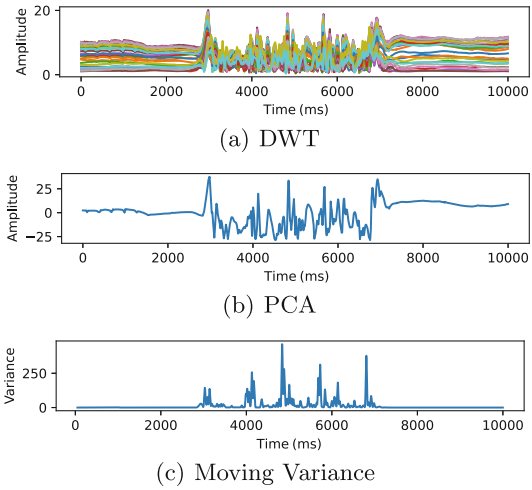
3.1 Overview

To realize online activity segmentation, we introduce a sliding window mechanism and ensure that all processing flows of activity segmentation are completed within the sliding window duration. As the overall workflow of LightSeg shown in Fig. 2, after obtaining a CSI stream online from a sliding window, LightSeg uses the preprocessing module to process the CSI stream into sensing data easy to segment, including compression, amplitude limiting, DWT, PCA, and moving variance calculation. Then the activity extraction module uses our core algorithm to determine the start and end points of the activities. LightSeg first determines the end of an activity and then decides the threshold suitable for the current activity granularity to get the CSI segment. The resulting CSI segment containing the activity will be used for the following activity classification. To facilitate comparison, we reproduce the model of DeepSeg as the classifier. Users can also design or use other activity classification models. The remaining CSI samples after the decided end point in the current window will be spliced into the next sliding window to avoid incomplete activity extraction.

3.2 Data Acquisition

To facilitate the online WiFi application, the CSI data stream should be carefully handled since the very beginning from data acquisition.

Online CSI Stream. We develop a CSI real-time parser to handle the online CSI data streams. For each pair of transceivers, each received packet contains the CSI of 30 OFDM subcarriers. The parser will store the real part of the CSI (i.e., CSI amplitude) in the queue and wait for the sliding window to obtain the data. For multiple transceiver links (e.g. 3 pairs of transceivers in our implementation), instead of using all the links, we just select one of the links with good signal



(a) DWT

(b) PCA

(c) Moving Variance

Fig. 3. Illustration of the preprocessing.

quality for activity segmentation and then obtain the segmented CSI data of multiple links.

Sliding Window. We use a sliding window to detect the target activities. Assume the length of a sliding window is $L_{sw} = L_{rest} + L_{new}$, where L_{rest} represents the remaining CSI in the previous sliding window and L_{new} represents the new CSI obtained from the queue in the current sliding window. When the queue length reaches L_{new}, the remaining CSI in the last segmentation will be combined with the CSI in the queue to form a new sliding window for segmentation, ensuring the integrity of the signal without overlapping. To prevent the length of the sliding window from being too long, resulting in substantial subsequent signal processing time, the rest length L_{rest} is limited and the rear of the remaining CSI signals will be retained if L_{rest} is larger than the limit. But if the limit of L_{rest} is too small, the unsegmented activity may be incomplete in the next sliding window; if it's too large, the processing time of the next sliding will increase. In our current implementation, we empirically set the limit to $1.5L_{new}$, which can avoid incomplete activities and minimize processing time.

Suppose the sampling rate of CSI is F_s, and the time of obtaining the new part of CSI equals L_{new}/F_s. For an online system, to avoid congestion caused by CSI in the queue that cannot be processed in time, the CSI of the current sliding window must be processed within L_{new}/F_s. In addition, sufficient time should be reserved for activity classification.

3.3 Preprocessing

The raw CSI data is extremely noisy and redundant. Extracting activities directly from CSI data is inaccurate and of high overhead. Hence, we design

a four-step preprocessing module to remove noise and redundancy, and provide data easy to process for the following activity extraction module.

Outlier Removal. Amplitude limiting can filter out the impulse outlier caused by accidental factors, such as the opening and closing of power switches. Because CSI amplitude is positive and usually less than 30, we set the CSI amplitude exceeding the interval [0,35] to the maximum or minimum value of the interval to reduce the impact of outliers on subsequent calculations.

Denoising. Discrete Wavelet Transform (DWT) [6] can achieve an obvious denoising effect and well retain the original characteristics of the signal. We perform DWT denoising on the CSI of 30 subcarriers, as shown in Fig. 3(a). The segmentation features can be enhanced after DWT noise filtering. In addition, due to the fast computation of DWT, it is suitable for online systems with the requirements of low latency.

Principal Component Analysis (PCA). The 30 OFDM subcarriers of a single antenna pair have strong linear correlation [12], so they contain a large number of redundant information [22], which can be removed during segmentation to reduce computing overhead. PCA is a widely used mathematical dimension reduction method [6,23] to remove the redundancy, which converts a group of linearly related variables into a group of linearly uncorrelated variables, and the output variables are arranged from large to small according to the retention ratio of signal characteristics. We select the first principal component for the further preprocessing and denote it as P, as shown in Fig. 3(b).

Moving Variance Extraction. As shown in Fig. 3(b), the CSI changes during the activity are larger than that during the static part. Hence, we use a linear data processing method, moving variance to capture the difference. Moving variance calculates the variance of P in sliding windows with a length of L_{mv} and the result is denoted as V, as shown in Fig. 3(c). The moving variance corresponding to i-th window is denoted as v_i. From Fig. 3(c), we can find that the moving variance of the activity is much larger than that of the static part. In addition, coarse-grained activities will have a larger moving variance. Figure 4 shows the moving variance result of the set of mixed-grained activities shown in Fig. 1. The results demonstrate that we can use moving variance as an indicator to learn the activity granularity.

Compression. LightSeg has short processing time ($\sim 0.3s$ per activity) in commonly used hardware such as laptops. However, in real scenarios, users may need

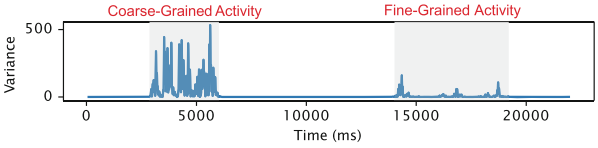


Fig. 4. Preprocessing results of mixed-grained activities.

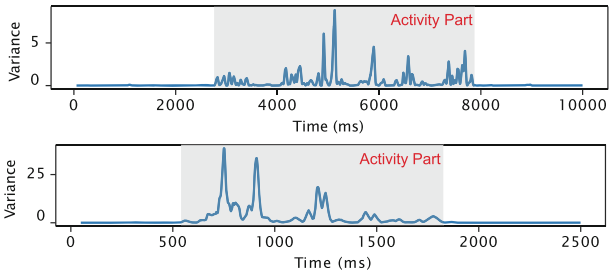


Fig. 5. Preprocessing results in different environments.

to deploy WiFi sensing to resource-limited hardware, which will increase the processing time of LightSeg and influence user experience. Reducing the sampling rate through data compression can significantly reduce the processing time, but it will sacrifice the segmentation accuracy instead. To balance the trade-off between efficiency and performance, a user can set the user-specified compression ratio. Since the low overhead of LightSeg, compression is only optional and not enabled by default. The compression method is the most commonly used equidistant sampling, which is consistent with DeepSeg.

Segmentation desires environment independence to be practical. Figure 5 shows the results V of activity "pushing" after above preprocessing in two different environments. It is obvious that results of the activity part in the two environment are very different. However, as long as the appropriate threshold is decided, the activity can be segmented because of the clear difference between the activity and the static part. Our threshold design is not predefined based on activity type or environment, it is adjusted based on each current activity. Hence, our method can be environment independent.

3.4 Activity Extraction

This module is designed to provide segmented CSI as the input for the classifier used for activity classification. The input size of the classifier is usually fixed, e.g., $200 \times 30 \times 3$. To retain the length feature of the activity, the length of all CSI segments is consistent. The segmented CSI can be fully retained or compressed according to the input size of the classifier. Suppose L_{output} is the output length of the activity extraction module, which is also the length of segmented CSI. It is usually larger than the length of the activity part, and the activity part will be placed in the middle of the segmented CSI.

To determine the position of the activity part, we design a lightweight algorithm described in Algorithm 1 to find the start and end points of the target activity. The inputs are the preprocessed CSI V, sliding window length L_{sw} and CSI output length L_{output}, and the outputs are the start and end points. The algorithm realizes granularity-aware threshold decision and abnormal peak removal. The former is used to perceive the granularity of the activity to segment, and the latter is used to resist interference.

Algorithm 1: Start and End Points Detection

Input: Sliding window length L_{sw}, CSI output length L_{output}, preprocessed CSI $V = \{v_1, ...v_2, ..., v_{L_{sw}}\}$

Output: Start point *Start* and end point *End* of the target activity

```
// find the end point of an activity first
```

$v_{max} = 0$, $End = 0$, $Start = 0$;
for $i = 0$; $i < L_{sw}$, $End == 0$; $i = i + 1$ **do**
  if $v_i > v_{max}$ **then**
    $v_{max} = v_i$;
    $\delta = 4\% \cdot v_{max}$;
  else
    if $v_i < \delta$ **then**
      for $a = i$; $a < i + L_{end}$, $\frac{number\ of\ v_a > \delta}{L_{end}} < 0.1$; $a = a + 1$ **do**
        if $\frac{number\ of\ v_a < \delta}{L_{end}} > 0.9$ **then**
          $End = i$;
          break;
      $i = a$;

```
// find the start point of the activity
```

for $i = End - L_{output}$; $i < End$, $Start == 0$; $i = i + 1$ **do**
  if $v_i > \delta$ **then**
    for $a = i$; $a < i + L_{end}$, $\frac{number\ of\ v_a < \delta}{L_{start}} < 0.5$; $a = a + 1$ **do**
      if $\frac{number\ of\ v_a > \delta}{L_{start}} > 0.5$ **then**
        $Start = i$;
        break;
    $i = a$;
return *Start*, *End*;

Granularity Aware Threshold Decision. For offline CSI stream processing, it is easy to control that only one activity exists in a sliding window or obtain the approximate location of a target activity. Then the threshold can be easily decided because only one activity is in the detection window. Based on the extensive experimental studies, we found that for a single activity in a detection window, setting the threshold to 4% of the maximum value of the target activity variance can accurately segment it. However, for online CSI streams, the time and number of activities occurrence are unpredictable, it is hard to ensure only one activity in each processing window (the sliding window). Then if more than one activities occur, the optimal threshold is hard to decide for all the activities, especially when they have different granularity.

To solve this problem, we propose a new threshold decision method, that is updating the maximum value online during the linear traversal of V, to obtain a granularity-aware threshold δ of the target activity. When the condition $v_i < \delta$ is continuously true for a certain number of samples, we find an end of an

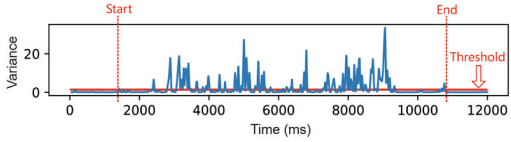
activity (line 9–11 in Algorithm 1). The number of samples is defined as the end window length L_{end} and the window is denoted as W_{end}. As shown in line 3–5 in Algorithm 1, the threshold δ keeps updating with the linear traversal of the current CSI steam. When the end of an activity is found, the corresponding threshold of this activity is decided. Then we can use this threshold to look for the start point.

Benefiting from linear traversal, the time complexity of our threshold decision is $O(N)$. Since V is positively related to activity granularity and the threshold is calculated from V, we can say our threshold has the ability to perceive and adapt to the granularity of the activity.

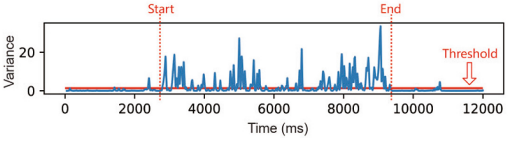
Abnormal Peak Removal. Due to pulse noise caused by environment changes or automatic gain control and even hardware errors of the WiFi transceivers, samples in the static part can be occasionally larger than the threshold δ, leading to incorrect detection of activities.

To cope with this problem, we propose abnormal peak removal that leverages the active duration correct the detection errors. We define the value of the static part above the threshold but with a duration obviously shorter than the duration of a normal activity as abnormal peak. From the concrete example in Fig. 6, we can find that the length of an abnormal peak is much shorter and its height is much smaller than the maximum value. Hence, we remove the peaks with a much short duration to calibrate the estimation of start and end points.

In the process of linear traversal of V, let current position i as the start of W_{end} and calculate the proportion of points smaller than δ in W_{end}. If the proportion exceeds a certain value, though some short-period peaks exist, i will be still regarded as the end of an activity. Otherwise, the activity does not end, W_{end} will be reset, and the start of the next W_{end} is the current position of the traversal. A similar operation is conducted when looking for the start point. By this way, we remove the abnormal peaks and obtain the correct start and end points of an activity. The result when using abnormal peak removal is shown in Fig. 6(b). Note that looking for the start point needs a back trace of the data



(a) Without abnormal peak removal.



(b) With abnormal peak removal.

Fig. 6. Segmentation results with and without abnormal peak removal.

but the traversal distance to find the start point is short. In general, the total distance of the twice traversal is less than L_{sw}.

In the algorithm, some parameters are determined empirically. L_{new} is set to the average length of normal activities, and adjustment within limits to it will not affect the segmentation performance. To ensure the integrity of the segmented CSI, we make L_{output} longer than the maximum length of the target activities. For different applications with different datasets, L_{new} and L_{output} can be different. But it should be noticed that all the system parameters are not affected by environment and activity granularity. The system parameters are general for different environments and activity granularity.

4 Evaluation

In this section, we first introduce the experiment settings and then present the experimental results comparing with baseline methods.

4.1 Experiment Settings

we evaluate LightSeg on the public dataset provided by DeepSeg [16], which contains both coarse-grained and fine-grained activities. The DeepSeg dataset [16] was collected in a meeting room. The transmitter is a commercial WiFi router with one antenna and the receiver is a laptop with Intel 5300 NIC and three antennas, which can provide CSI of 30 subcarriers from each pair of transmit-receive antennas. The dataset was provided by five users with different body shapes and ages, each of whom performed 10 activities 30 times each, including 5 coarse-grained activities (boxing, picking up, running, squatting and walking) and 5 fine-grained activities (hand swing, hand raising, pushing, drawing O and drawing X). The sampling rate of the collected data is 1000 packets/s. Based on these data, 150 activity sequence traces are used. In each CSI sequence, 10 activities were performed by a user, including a coarse-grained activity 5 times and a fine-grained activity 5 times. 80% of the traces are used as the training set and the rest 20% traces are used as the test set.

Based on the real-time data processing method in MATLAB developed by Lu et al. [4], we developed a CSI parser in Python that can run online according to the experimental requirements. To evaluate the online segmentation process, we develop a CSI data sending tool based on Python, which can send the CSI trace in DAT format to the CSI parser at a user-specific sending rate.

For comparison, we select DeepSeg [16], the state-of-the-art method that achieves the highest segmentation accuracy as the baseline of deep learning-based methods, and Wi-Multi [3] as the baseline of threshold-based segmentation methods. It's worth noting that DeepSeg reduces the sampling rate from 1000 to 50 packets/s because of the unaffordable high computation overhead. But for the threshold-based methods, Wi-Multi and LightSeg, there is no need to down sampling due to the much lower computation overhead.

We use the segmentation accuracy metric mentioned in DeepSeg to measure the performance of segmentation methods. For a given activity, the segmentation accuracy is equal to $|A \cap B|/max\{|A|, |B|\}$, where A represents the ground truth set of the start and end points of the activity, the start and end points of all activities are labeled and published by DeepSeg, and B represents the set of start and end points predicted by the segmentation method.

4.2 Performance of Activity Segmentation

We first compare LightSeg with DeepSeg and Wi-Multi and then evaluate its performance under different settings.

Comparison with DeepSeg. We first compare LightSeg with DeepSeg, which also aims at solving the granularity problem of activity segmentation for WiFi sensing. We compare their segmentation accuracy in three scenarios, (1) mixed-grained scenario where 5 coarse-grained and 5 fine-grained activities appear alternately, (2) coarse-grained scenario where only coarse-grained activities exist, and (3) fine-grained scenarios where only fine-grained activities exist. In all the scenarios, similar to the setting of DeepSeg, each activity sequence trace contains 10 activities. Then we use different methods to segment all the activities.

The average segmentation accuracy is shown in Fig. 7. We can find that in all the three scenarios, LightSeg achieves the best performance. The accuracy of LightSeg is 93%, 92.5%, and 93.6% in the mixed-grained, coarse-grained, and fine-grained scenarios, which is even slightly higher than DeepSeg. This is because LightSeg can perform segmentation on the high-rate CSI data but DeepSeg has to run on the down-sampling data, which inevitably causes inaccuracy. To validate this, we also conduct LightSeg on the data with the same sampling rate of DeepSeg and plot its results as the LightSeg with Compression in Fig. 7. We can find that when running on the compressed data, LightSeg has obvious performance degradation and its accuracy in three scenarios is 3.7% lower than DeepSeg in average. The result is as expected because with the same data, deep learning technique is much more powerful to extract segmentation

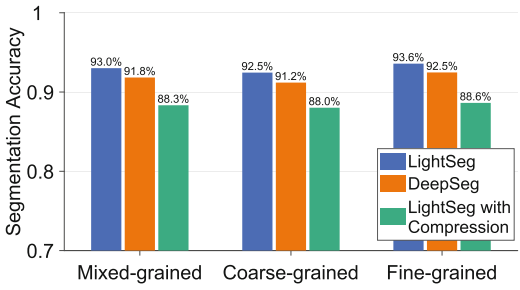


Fig. 7. Comparison with DeepSeg

features than the threshold. But thanks to the low overhead, threshold-based methods can use the high-rate CSI data to achieve better accuracy.

Comparison with Wi-Multi. We then compare LightSeg with Wi-Multi, the existing threshold-based method. Wi-Multi dynamically decides the threshold according to the parameter called sampling rate which must be preset according to the application scenario. Here, the so-called sampling rate in Wi-Multi means the ratio of the CSI sequence length to the slot length, i.e., the slot numbers in a detection window. It is different from the signal sampling rate. To avoid confusion, we use R_s to represent the sampling rate parameter in [3]. By default, Wi-Multi set R_s as 80. Given R_s, Wi-Multi can adjust the threshold according to the activities in the detection window. Hence, it still has the granularity problem. R_s has to be preset for different granularity scenarios, which is impossible for the online systems that cannot predict what activity is coming.

We vary R_s to investigate the accuracy of Wi-Multi under different granularity. The results are shown in Fig. 8. We can find that the segmentation accuracy in three scenarios reaches the peak at different R_s, which reveals that different granularity activities correspond to different optimal R_s. However, in practice, we cannot know which activities we are dealing with in the current detection window before classification. It is hard to preset or dynamically adjust R_s online. But LightSeg has no such problem because it leverages the granularity-aware threshold for the current activity.

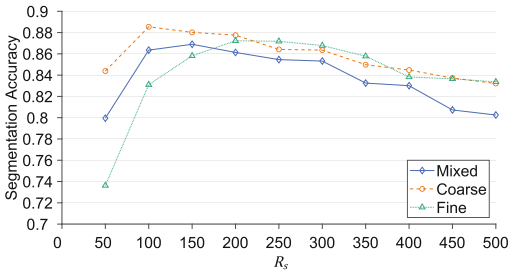


Fig. 8. Wi-Multi with different R_s

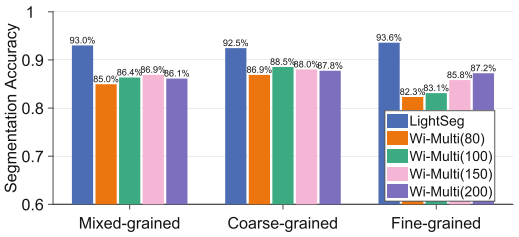


Fig. 9. Comparison with Wi-Multi

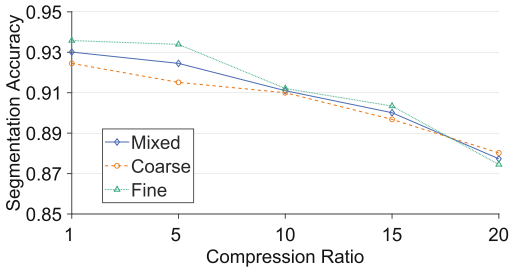


Fig. 10. LightSeg with different compression ratio

We compare the performance of Wi-Multi and LightSeg. Besides the default setting in [3], Wi-Multi(80), we also select several representative R_s for Wi-Multi according to the results in Fig. 8, including the mixed-grained optimal value 150, the coarse-grained optimal value 100, the fine-grained optimal value 200. The segmentation accuracy of LightSeg and Wi-Multi with different R_s is shown in Fig. 9. We can find that LightSeg has superior performance than Wi-Multi for all the three scenarios. Wi-Multi(80), the default setting, has the lowest accuracy, indicating the setting suitable for the dataset in [3] is no longer appropriate to the DeepSeg dataset. For mixed-grained, coarse-grained, fine-grained activities, Wi-Multi with the corresponding optimal setting has the best performance among the four settings, which is 86.9%, 88.5%, and 87.2% respectively for the three scenarios. Compared to the optimal result of Wi-Multi, LightSeg can still achieve 7.0%, 4.5%, and 7.3% higher accuracy. In practice, with a fixed preset R_s, LightSeg can obtain even better performance improvement.

Impact of Data Compression. We also investigate the impact of data compression on segmentation. When enabling compression, the CSI trace will be first down sampled and then performing segmentation. Figure 10 shows the accuracy degradation when increasing the compression ratio. When increasing the compression ratio from 1 to 20, the accuracy drops by 5% in average. When the compression ratio is 15, i.e., the sampling rate is 66 packets/s, LightSeg can still achieve the accuracy higher than 90%. Actually, due to the low overhead, LightSeg can directly use the data with the original sampling rate (1000 packets/s) and even higher rates.

Impact of the Selected Antenna. Unlike DeepSeg, which uses all receiving antennas as input, LightSeg selects one of the receiving antennas with good signal quality (Rx-1) for segmentation and avoids unnecessary computation. Figure 11 shows the segmentation accuracy of LightSeg when using Rx-1 and all antennas. Surprisingly, we find that using all antennas can even lower the accuracy. This is because there is an antenna providing low-quality CSI data that brings interference to the segmentation.

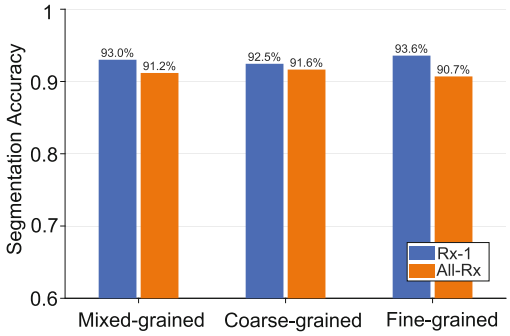


Fig. 11. LightSeg with Rx-1 or All-Rx

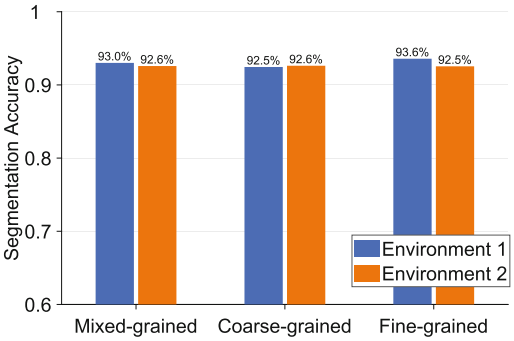


Fig. 12. Segmentation performance in different environments.

Environment Independence. To verify the environment independence of LightSeg, we collect a new dataset in a campus lab and evaluate the segmentation performance. Both the transmitter and receiver are industrial computers equipped with Intel 5300 NIC. The transmitter and receiver enable one and three antennas respectively. The sampling rate is 1000 packets/s. The dataset contains 1080 activity CSI traces provided by six users with different body shapes. Each user performs 9 activities 20 times each, including 4 coarse-grained daily activities (bending, falling, sitting and lying down) and 5 fine-grained gestures (pushing, sliding up, sliding up, sliding down, sliding left and sliding right).

Figure 12 presents the segmentation accuracy in two different environments, where Environment 1 represents the experimental environment of DeepSeg's dataset and Environment 2 represents the experimental environment of our dataset. For the results, we can clearly find that although the experimental environment is changed, LightSeg achieves similar performance for all the three activity granularity scenarios. The results demonstrate that LightSeg is environment independent.

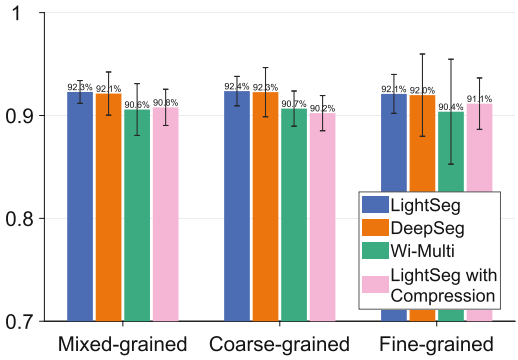


Fig. 13. Accuracy of activity classification

4.3 Performance of Activity Classification

We further evaluate the impact of segmentation on activity classification performance. We use the CNN-based deep learning model publicly provided by DeepSeg. The network structure of the CNN model used as the classifier is shown in Table 1 and the hyper-parameters of the classifier are consistent with the parameters used in DeepSeg. We use 80% of the CSI traces in DeepSeg dataset as the training set and the other 20% of the traces as the test set. We repeat the experiment ten times and plot the accuracy of four methods in Fig. 13, where the error bars indicate standard deviation.

Table 1. Network Architecture of CNN

NO	Operation	Configuration
1	Input	$200 \times 30 \times 3$ (CSI values)
2	Dropout	p = 0.2
3	Conv	Kernel = 3×3 FM = 96 WN lReLU
4	Conv	Kernel = 3×3 FM = 96 WN lReLU
5	Conv	Kernel = 3×3 Stride = 4×2 FM = 96 WN lReLU
6	Dropout	p = 0.5
7	Conv	Kernel = 3×3 FM = 192 WN lReLU
8	Conv	Kernel = 3×3 FM = 192 WN lReLU
9	Conv	Kernel = 3×3 Stride = 4×2 FM = 192 WN lReLU
10	Dropout	p = 0.5
11	Conv	Kernel = 3×3 FM = 192 WN lReLU
12	NiN	FM = 192 WN IReLU
13	NiN	FM = 192 WN IReLU
14	Pooling	FM = 192 Global Pooling
15	Dense	FM = 4 WN

Thanks to the high segmentation accuracy, LightSeg achieves the best classification accuracy, as expected. Overall, the classification accuracy is consistent to the segmentation performance, which proves that accurate activity segmentation indeed helps improve the classification accuracy. In three scenarios with different granularity, the classification accuracy of LightSeg is similar to DeepSeg, and 1.9% higher than Wi-Multi in average.

4.4 Evaluation of Computation Overhead

LightSeg is expected to have a low computation overhead due to our sophisticated designs. We evaluate the overhead of each step of LightSeg , compared to DeepSeg. Table 2 shows the hardware platform used for evaluation, including Raspberry Pi as the resource-limited hardware, a laptop as the normal WiFi device, and a high-performance server equipped with GPU resources.

Table 2. Hardware Information

	Raspberry Pi	Laptop	Server
Model	Raspberry Pi 4B	HUAWEI MateBook 14	DELL PowerEdge
CPU	Broadcom BCM2711	AMD Ryzen 4800H	Intel Xeon 6138
GPU	Null	Null	Nvidia GTX 2080Ti
Memory	4 GB	16 GB	256 GB
OS	Linux Debian 10	Windows 10	Linux Ubuntu 16.04

Table 3 shows the evaluation result of LightSeg. For the laptop using the original sampling rate (1000 packets/s), the average processing time of each activity is $0.3s$, which is enough to meet the latency requirement of online activity segmentation. In addition, the maximum memory used is 193 MB, which is affordable for most common devices. For the Raspberry Pi using the original sampling rate, the total processing time is $3.48s$, which is much larger. Hence, when applying WiFi sensing on a resource-limited device, we suggest using down-sampling data for segmentation. When using the rate of 50 packets/s, the total processing time is reduced to $0.2s$ and the used memory is about 100 MB. Although the sampling rate is reduced, the segmentation accuracy and classification accuracy still remain above 88% and 90% as shown in Fig. 7 and Fig. 13.

The input data of DeepSeg for each processing contains ten activities. We evaluate the average processing time of each activity and the memory used during the test phase of activity segmentation (excluding model training). Table 4 shows the results. The first two steps of DeepSeg run in MATLAB. Although MATLAB has a faster processing speed than Python, it uses more memory. For state inference, we run it on both the laptop and the server. The processing time is $73.92s$ and $23.35s$ when running on the laptop and the server. The average processing time for one activity is $9.976s$ when using CPU only and is $4.919s$ even when using GPU. But using GPU requires a much larger memory. Under the same hardware condition with the laptop, the average processing time of LightSeg is only 3% of that of DeepSeg.

Table 3. Computation overhead of LightSeg

Hardware	Laptop	Raspberry Pi	Raspberry Pi
Sampling Rate (packets/s)	1000	1000	50
Maximum Memory Used	193 MB	225 MB	103 MB
Preprocessing step	Processing Time (s)		
Outlier Removal	0.0031	0.0035	0.0007
DWT	0.2096	0.8695	0.0093
PCA	0.0403	1.3621	0.0237
Moving Variance	0.0016	0.0037	0.0022
Find End Point	0.0375	0.8292	0.0560
Find Start Point	0.0173	0.4183	0.0221
Total	0.3094	3.4862	0.1976

Table 4. Overhead of DeepSeg

Step	Hardware	Runtime Environment	Processing Time (s)	Memory	Graphics Memory
Preprocessing	Laptop	MATLAB	10.09	2901 MB	
Discretize CSI			15.62	2496.8 MB	
State Inference		Python	73.92	1449 MB	
	Server		23.35	3.803 GB	4474 MB
Extract Activity	Laptop		0.13	31.4 MB	
Total			99.76 (CPU) 49.19 (CPU+GPU)		
Average			9.976 (CPU) 4.919 (CPU+GPU)		

5 Conclusion

We present a low-latency activity segmentation method LightSeg for WiFi sensing, which can segment activities of different granularity online. We design a granularity aware threshold that can be quickly adjusted according to the activity granularity, which solves the performance degradation of mixed-grained activity segmentation. The algorithm has low complexity and high segmentation accuracy. We deploy LightSeg in hardware with different performance and evaluated the segmentation accuracy and overhead. Experimental results demonstrate the effectiveness and low overhead of LightSeg.

References

1. Bu, Q., Yang, G., Ming, X., Zhang, T., Feng, J., Zhang, J.: Deep transfer learning for gesture recognition with Wi Fi signals. Pers. Ubiquit. Comput., 1–12 (2020)
2. Chen, L., Chen, X., Ni, L., Peng, Y., Fang, D.: Human behavior recognition using Wi-Fi CSI: challenges and opportunities. IEEE Commun. Mag. **55**(10), 112–117 (2017)

3. Feng, C., Arshad, S., Zhou, S., Cao, D., Liu, Y.: Wi-multi: a three-phase system for multiple human activity recognition with commercial WiFi devices. IEEE Internet Things J. **6**(4), 7293–7304 (2019)
4. Lu, B., Zeng, Z., Wang, L., Peck, B., Qiao, D., Segal, M.: Confining Wi-Fi coverage: a crowdsourced method using physical layer information. In: Proceedings of IEEE SECON (2016)
5. Ma, Y., Zhou, G., Wang, S.: WiFi sensing with channel state information: a survey. ACM Comput. Surv. (CSUR) **52**(3), 1–36 (2019)
6. Palipana, S., Rojas, D., Agrawal, P., Pesch, D.: FallDefi: ubiquitous fall detection using commodity Wi-Fi devices. Proc. ACM Interact., Mob., Wearable Ubiquit. Technol. **1**(4), 1–25 (2018)
7. Sheng, B., Xiao, F., Sha, L., Sun, L.: Deep spatial-temporal model based cross-scene action recognition using commodity WiFi. IEEE Internet Things J. **7**(4), 3592–3601 (2020)
8. Virmani, A., Shahzad, M.: Position and orientation agnostic gesture recognition using WiFi. In: Proceedings of ACM MobiSys (2017)
9. Wang, F., Gong, W., Liu, J.: On spatial diversity in WiFi-based human activity recognition: a deep learning-based approach. IEEE Internet Things J. **6**(2), 2035–2047 (2018)
10. Wang, H., Zhang, D., Wang, Y., Ma, J., Wang, Y., Li, S.: RT-fall: a real-time and contactless fall detection system with commodity WiFi devices. IEEE Trans. Mob. Comput. **16**(2), 511–526 (2016)
11. Wang, W., Liu, A.X., Shahzad, M.: Gait recognition using WiFi signals. In: Proceedings of ACM UbiComp (2016)
12. Wang, W., Liu, A.X., Shahzad, M., Ling, K., Lu, S.: Understanding and modeling of WiFi signal based human activity recognition. In: Proceedings of ACM MobiCom (2015)
13. Wang, Z., Guo, B., Yu, Z., Zhou, X.: Wi-Fi CSI-based behavior recognition: from signals and actions to activities. IEEE Commun. Mag. **56**(5), 109–115 (2018)
14. Wu, D., Zhang, D., Xu, C., Wang, H., Li, X.: Device-free WiFi human sensing: from pattern-based to model-based approaches. IEEE Commun. Mag. **55**(10), 91–97 (2017)
15. Wu, X., Chu, Z., Yang, P., Xiang, C., Zheng, X., Huang, W.: TW-see: human activity recognition through the wall with commodity Wi-Fi devices. IEEE Trans. Veh. Technol. **68**(1), 306–319 (2018)
16. Xiao, C., Lei, Y., Ma, Y., Zhou, F., Qin, Z.: DeepSeg: Deep-learning-based activity segmentation framework for activity recognition using WiFi. IEEE Internet Things J. **8**(7), 5669–5681 (2020)
17. Xu, L., Zheng, X., Li, X., Zhang, Y., Liu, L., Ma, H.: WiCAM: imperceptible adversarial attack on deep learning based Wi-Fi sensing. In: Proceedings of IEEE SECON (2022)
18. Yan, H., Zhang, Y., Wang, Y., Xu, K.: WiAct: a passive WiFi-based human activity recognition system. IEEE Sens. J. **20**(1), 296–305 (2019)
19. Yang, K., Zheng, X., Xiong, J., Liu, L., Ma, H.: WiImg: pushing the limit of WiFi sensing with low transmission rates. In: Proceedings of IEEE SECON (2022)
20. Yang, Z., Zhou, Z., Liu, Y.: From RSSI to CSI: indoor localization via channel response. ACM Comput. Surv. (CSUR) **46**(2), 1–32 (2013)
21. Yousefi, S., Narui, H., Dayal, S., Ermon, S., Valaee, S.: A survey on behavior recognition using WiFi channel state information. IEEE Commun. Mag. **55**(10), 98–104 (2017)

22. Zhang, L., Liu, M., Lu, L., Gong, L.: Wi-Run: multi-runner step estimation using commodity Wi-Fi. In: Proceedings of IEEE SECON (2018)
23. Zhang, L., Wang, C., Ma, M., Zhang, D.: WiDIGR: direction-independent gait recognition system using commercial Wi-Fi devices. IEEE Internet Things J. **7**(2), 1178–1191 (2019)
24. Zhang, L., Zhang, Y., Zheng, X.: WiSign: ubiquitous American sign language recognition using commercial Wi-Fi devices. ACM Trans. Intell. Syst. Technol. (TIST) **11**(3), 1–24 (2020)
25. Zheng, X., Wang, J., Shangguan, L., Zhou, Z., Liu, Y.: Smokey: ubiquitous smoking detection with commercial WiFi infrastructures. In: Proceedings of IEEE INFOCOM (2016)
26. Zheng, X., Wang, J., Shangguan, L., Zhou, Z., Liu, Y.: Design and implementation of a CSI-based ubiquitous smoking detection system. IEEE/ACM Trans. Networking **25**(6), 3781–3793 (2017)