Distance Functions to Detect Changes in Data Streams

Ulziitugs Bud*, and JongTae Lim*

Abstract: One of the critical issues in a sensor network concerns the detection of changes in data streams. Recently presented change detection schemes primarily use a sliding window model to detect changes. In such a model, a distance function is used to compare two sliding windows. Therefore, the performance of the change detection scheme is greatly influenced by the distance function. With regard to sensor nodes, however, energy consumption constitutes a critical design concern because the change detection scheme is implemented in a sensor node, which is a small battery-powered device. In this paper, we present a comparative study of various distance functions in terms of execution time, energy consumption, and detecting accuracy through simulation of speech signal data. The simulation result demonstrates that the Euclidean distance function has the highest performance while consuming a low amount of power. We believe our work is the first attempt to undertake a comparative study of distance functions in terms of execution time, energy consumption, and accuracy detection.

Keywords: change detection, distance functions.

1. Introduction

In recent years, detecting changes in data streams has become one of the critical issues concerning the various applications including environmental (*monitoring environmental conditions*), health (*integrated patient monitoring*), home (*local and remote management of home devices*), military, and other commercial applications. In these applications, data is generated by the underlying process changed over time. Many critical decisions depend on the accuracy of the detected changes.

The purpose of change detection is to cope with an abnormal status very quickly. Several approaches have emerged, the most popular of which is the sliding window model [1, 2]. In the sliding window model, a distance function is used to detect changes in data streams by comparing two sliding windows. Therefore, choosing the distance function is of critical importance.

In this paper we present a comparative study of various distance functions in terms of execution time, energy consumption, and accuracy detecting when data is changed both gradually and precipitously. When estimating the execution time and energy consumption, we assume that the detection change scheme would be implemented as embedded software in a 32-bit RISC processor. Then we use the instruction level power estimation tool Joule Track [3]. This tool has been proposed to compute the energy consumption of given software on a StrongARM processor, which is one of the most popular embedded processors used in the sensor network [4]. The rest of this paper is organized as follows. Works related to change detection schemes are explained in Section 2. Section 3 introduces our own experimental change detection scheme. Section 4 explains the results of the experiments. Finally, Section 5 offers the conclusions.

Manuscript received February 11, 2006; accepted March 3, 2006.
Corresponding Author: JongTae Lim
* Department of Computer Science, Kongju National University, Kongju, Korea. (b_ulziitugs@yahoo.com, jtlim@kongju.ac.kr)

2. Related works

We have surveyed related research in two areas: 1) study of change detection schemes; and 2) study of distance functions. The paper by Kifer, Ben-David, and Gehrke [1] lays out a comprehensive non-parametric framework for change detection. They exploit the order statistics of the data, and define the generalizations of the Wilcoxon and Kolmogorov-Smirnoff test in order to define the distance between two distributions. Their method is effective for one-dimensional data streams; however, as they point out, their approach does not significantly extend to higher dimensions. In their paper [2], Tamraparni, Shankar, and Suresh propose an information-theoretic approach to change detection. The approach is based on the Kullback-Liebler distance function and the statistical method of bootstrapping to provide a formal probabilistic guarantee for change detection. This approach is non-parametric; it requires no assumptions regarding the data streams. Also, it has been shown to be effective for a variety of multidimensional data sets with different notions of change. Thus, a variety of change detection schemes have been proposed and summarized in [5-9].

In contrast, there are few works which focus on the study of distance functions. Some studies have been undertaken, but they only have studied distance function from the viewpoint of information or mathematical theory [10]. Indeed, the change detection schemes are usually implemented as embedded software on a battery-powered

sensor node [11-13]. Therefore it is valuable to study distance functions in term of execution time, energy consumption, and accuracy detection, and we believe our work is the first attempt to show such a comparative study of distance functions.

3. Change detection scheme

In this section, we describe our experimental change detection scheme. We consider that a data stream is a sequence of data such as $\{x_1, x_2 \ldots .\}$ and that each x_i item is generated independently by distribution. In the experimental scheme, the sliding window model is used to detect changes in data streams. Fig. 1 shows a sliding window model. One advantage of using the sliding window model is that it reduces the persistent problem of detecting changes over data streams to the problem of comparing samples in two windows that are generated by different distributions.

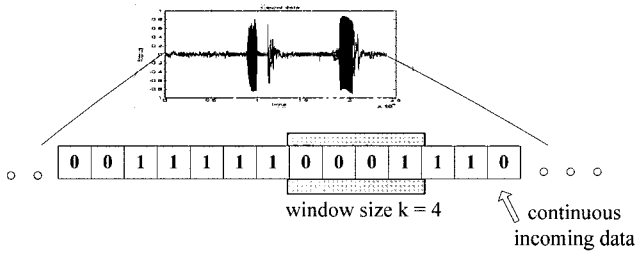

Fig. 1. Sliding window model

There are 2 kinds of sliding window model such as: 1) the adjacent window model; 2) and the fix-slide window model. In Fig. 2, the basic processes of these 2 kinds of sliding window model are described.

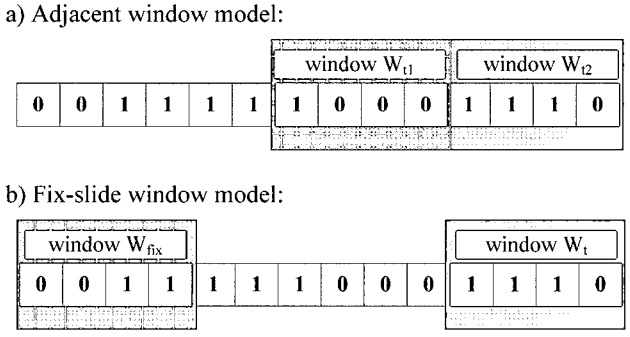

Fig. 2. Two kinds of sliding window model

In the adjacent window model, two adjacent windows, W_{t1} and W_{t2}, are compared by a distance function. First, the two windows are compared, then the window W_{t1} data is updated with the $W_{t2\ data}$ and the window W_{t2} data is updated by the incoming data stream. In the fix-slide window model, a sliding window W_{t1} and a fixed window W_{fix} are compared by the distance function. First, the two windows are compared, and if there is reasonable change between the two windows, then the window W_{fix} data is updated with the window W_t data, and the window W_t data is updated by the incoming data stream. Otherwise, only the window W_t data is updated by the incoming data stream. According to each sliding window model's features the adjacent window model detects changes better when the data is changed precipitously. By contrast, the fix-slide window model is more suitable for detecting changes when the data changes gradually.

Algorithm 1 describes the change detection algorithm with the adjacent window model in detail.

Algorithm1: Change detection (adjacent window model)
Step1: **for** i = 1 . . . k **do**
t = 0
construct window W_{t2}
end for
Step2: **while** not at end of stream **do**
for i = k . . . 2k **do**
$W_{t1} = W_{t2}$ (old data)
construct W_{t2}
if $d(W_{t2}, W_{t1}) > d^*$
t = current time
report change at time "t"
GOTO step 1
else if GOTO step 2
end if
end for
end while

In algorithm 1, k equals window size, d represents distance function, and d* is a balance value between sensitivity and robustness of the detection. We chose an arbitrary window size in our simulation. There is no obvious method for choosing the effective window size, and this constitutes a big challenge in this design area.

We carried out a simulation on several distance functions, including: 1) Euclidean; 2) Manhattan; 3) Lance and Williams; 4) Quadratic; 5) Bhattacharya; 6) Kullback-Leibler; and 7) Delta. A brief description of the Euclidean distance function is illustrated in equation (1).

$$d(p,q) = \sqrt{\frac{1}{n} * \sum_{i}^{n} (p(i) - q(i))^2} \qquad (1)$$

, where n is window size, and p(x) and q(x) are the data streams generated by some distributions. Other descriptions of the distance function are summarized in [10] and some features of each distance function are described by the simulation results in Section 4.

4. Simulation results

In this section, we summarize our simulation results. Our experiments were performed on the Intel Pentium IV 2.8 GHz machine with a single processor and 512 MB of physical memory. Fig. 3 shows the main framework of the

simulation environment. We chose a speech signal as a data stream. Therefore, in this virtual simulation environment, the sensor is the microphone, the sink is the sound card, and the base station is the host PC.

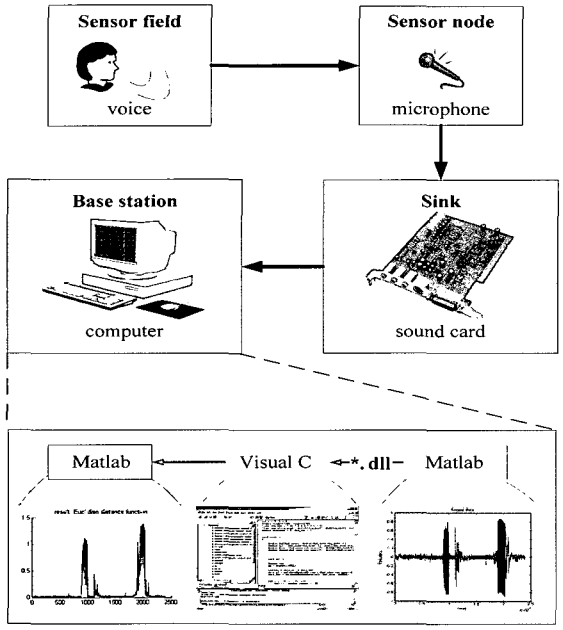

Fig. 3. Framework for experimental change detection scheme

Sound is converted into electrical current through the microphone. This fast changing voltage is converted into a series of numbers by the sound card and is then sent to the base station. The primary simulation task is performed through two main sub processes: data capturing from the sound card is performed by calling the Matlab dll (dynamic link library) functions, while the process of detecting changes in the captured data is performed through our change detection scheme, which is implemented with Visual C++ language.

Here, the Matlab dll library includes our predefined functions that are used to execute the Fourier transformation on the capturing speech signal. It also outputs the frequency stream of the speech signal. The change detection scheme detects changes in this stream, after which the detection results are sent to the Matlab application through the Matlab & C/C++ interface functions, where they are analyzed and visualized. Such a simulation framework provides a very efficient, flexible, and designer-friendly environment. In particular, we have the ability to conduct various analyses and comparisons using the Matlab functions on the various kinds of distance functions and schemes of detection change.

Moreover, we assume that the change detection scheme may be implemented as embedded software on a 32-bit RISC processor such as StrongARM. In such a case, the estimation of energy consumption and the execution time of the change detection process must be measured on the processor.

We estimated the energy consumption and the execution time of each distance function by using an instruction level power estimation tool called Joule Track [3]. That tool has been proposed to compute the energy consumption of given software on the ARM processor. Fig. 4 shows the execution times and Fig. 5 shows the energy consumption of the distance functions, respectively.

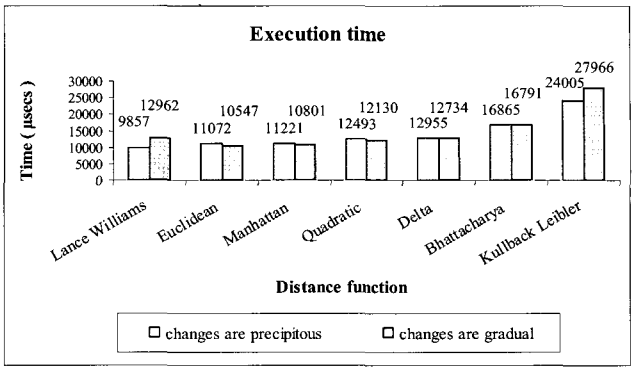

Fig. 4. Execution time of distance functions

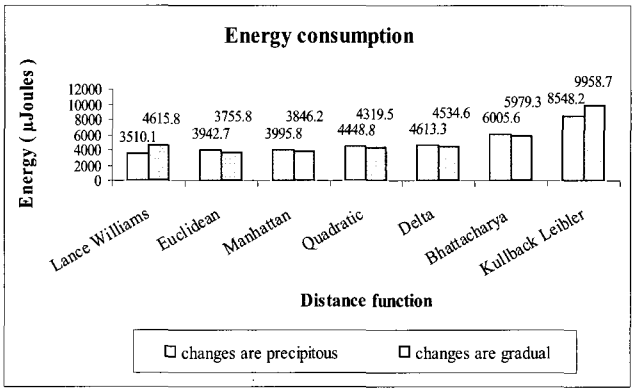

Fig. 5. Energy consumption of distance functions

Our next trade-off between distance functions is performed by changing the window sizes of the scheme into two classes: gradual and precipitous.

Table 1 summarizes the simulation results. The window size appears to be a crucial parameter. If the window size is too large, then small changes will be drowned out. If it is too small, spurious changes will be detected, or possibly even missed altogether. So we perform a simulation of each distance function in the three window sizes of 50, 250, and 500 points. In the given simulation environment, data stream consists of N = 80000 points, with change occurring every 400 points. From the simulation results, the Euclidean and the Kullback-Leibler distance functions are better than the others in terms of detection. However, the execution time and the energy consumption of the Kullback-Leibler are 2.5 times greater than those of the Euclidean distance function, as shown in figures 4 and 5. This means that if the rate of the incoming data stream on a sensor is high, then the Kullback-Leibler distance function will perform more slowly than the Euclidean distance function, while the percentage of late data measurement will be greater than the Euclidean distance function. Due to such shortcomings, we preferred the Euclidean distance function.

Table 1. Comparison of distance functions

Distance function	Window size	Precipitous change		Gradual change	
		Detected (%)	Missed (%)	Detected (%)	Missed (%)
Kullback Leibler	50	100	0	100	0
	250	89.21	10.79	63.16	36.84
	500	51.58	48.42	42.11	57.89
Euclidean	50	94.74	5.26	100	0
	250	89.47	10.53	63.16	36.84
	500	52.63	47.37	42.11	57.89
Delta	50	84.21	15.79	57.89	42.11
	250	73.68	26.32	47.37	52.63
	500	42.11	57.89	42.11	57.89
Quadratic	50	78.95	21.05	94.74	5.26
	250	57.89	42.11	57.89	42.11
	500	42.11	57.89	42.11	57.89
Manhattan	50	52.63	47.37	94.74	5.26
	250	47.37	52.63	63.16	36.84
	500	26.32	73.68	42.11	57.89
Bhatta-charya	50	42.11	57.89	52.63	47.37
	250	36.84	63.16	52.63	47.37
	500	31.58	68.42	42.11	57.89
Lance and Williams	50	15.79	84.21	10.53	89.47
	250	0	100	0	100
	500	0	100	0	100

5. Conclusion

This paper presents a comparative performance study of several distance functions including Bhattacharya, Kullback-Leibler, Euclidean, Manhattan, Delta, Quadratic, and Lance and Williams in terms of execution time, energy consumption, and accuracy detection. The simulation results demonstrate that the Euclidean distance function has the highest performance while consuming the lowest power.

Reference

[1] D.Kifer, S.Ben-David, and J.Gehrke. "Detecting change in data streams" In VLDB, 2004.

[2] T.Dasu, S.Krishnan. "An information-theoretic approach to detecting changes in multi-dimensional data streams", AT&T Labs Research, Duke Uni., 2005.

[3] A.Sinha, A.P.Chandrakasan, "JouleTrack - A web based tool for software energy profiling", Massachusetts Institute of Technology, 2001.

[4] http://www.intel.com/design/strong/applnots/sa1100lx

[5] S.S.Chawathe, H.GarciaMolina. "Meaningful change detection in structured data" In SIGMOD, 1997.

[6] Y.Zhu, D.Shasha. "Efficient elastic burst detection in data streams" In KDD, 2003.

[7] V.Ganti, J.Gehrke, R.Ramakrishnan, and W.Y.Loh. "A framework for measuring differences in data characteristics" Computer and System Sciences, 2002.

[8] G.Hulten, L.Spencer, and P.Domingos. "Mining time changing data streams" In KDD, 2001.

[9] C.C.Aggarwal. "A framework for diagnosing changes in Evolving Data Streams" In SIGMOD, 2003.

[10] Rui Xu, Donald Wunsch, "Survey Of Clustering Algorithms", IEEE Transactions On Neural Networks, Vol. 16, No. 3, May 2005.

[11] Rex Min, Manish Bhardwaj, Seong-Hwan Cho, Nathan Ickes, Eugene Shih, Amit Sinha, Alice Wang, and Anantha Chandrakasan, "Energy-Centric Enabling Technologies For Wireless Sensor Networks" IEEE Wireless Communications, August 2002.

[12] Amit Sinha, Alice Wang, and Anantha Chandrakasan "Energy Scalable System Design" IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 10, No. 2, April 2002.

[13] Amit Sinha Anantha Chandrakasan "Dynamic Power Management In Wireless Sensor Networks" IEEE Design & Test Of Computers, 2001.

Ulziitugs Bud

She received a BS in Electronics Engineering from the Mongolian National University in 2002. She is now undertaking a master's course at Kongju National University in Korea. Her research interests include sensor networks, embedded SW design, and low power design.

JongTae Lim

He received a Ph.D. degree in Computer Science from KAIST in 1992. He has been a professor at Kongju National Univ. since 1993. His research interests are in the area of Information retrieval, Sensor Networks, Database Systems, Bioinformatics, and Simulation.