Energy Characterization and Optimization of Embedded Data Mining Algorithms: A Case Study

Hanqing Zhou, Lu Pu, Yu Hu, Xiaowei Xu *
School of Optical Electronic Information
Huazhong University of Science and Technology
Wuhan, China
xuxiaowei@hust.edu.cn

Aosen Wang, Wenyao Xu
Department of Computer Science & Engineering
University at Buffalo, the State University of New York
New York, USA
wenyaoxu@buffalo.edu

***Abstract*—Data mining has been flourishing in the information-based world. In data mining, the DTW-kNN framework is widely applied for classification in miscellaneous application domains. Most of the studies in the DTW-kNN framework focus on accuracy and speedup. However, with increasingly emphasis on applications of mobile and embedded systems, energy efficiency becomes an urgent consideration in data mining algorithm design. In this paper, we present our work on energy characterization and optimization of data mining algorithms. Through a case study of the DTW-kNN framework, we investigate multiple existing strategies to improve the energy efficiency without any loss of algorithm accuracy. To the best of our knowledge, this is the first work about energy characterization and optimization of data mining algorithms on embedded computing testbeds. All the experiments are implemented on a developed energy measurement testbed. The experimental results indicate that the distance matrix calculation is the bottleneck of the DTW-kNN framework, which accounts for 89.14% on average of the total energy. With several optimized methods, the reduction of the total energy in the DTW-kNN framework can reach as much as 74.6%.**

Keywords*—*Dynamic Time Warping; k-Nearest Neighbors; energy optimization; embedded computing testbed

I. Introduction

In the past several decades, with the explosion of information, data mining has become an increasingly attractive topic, which is used to discover implicitly and potentially useful knowledge from data [1]. With the approaching of information technology, the popularization of internet and the development of mobile internet, data mining has been flourishing and utilized in every regarding human life [2].

For applications in internet of things (IOT) and mobile internet, data mining becomes challenging in the following reasons. First, the scale of data in IOT and mobile internet is growing very fast, which makes it difficult to store all the data for analysis [2]. Second, data transmission in wireless sensor networks and wearable technology is based on wireless communication, which is rather energy-expensive to transmit all the sensing data [3]. Third, many applications (e.g. medical applications and financial market applications) require quick action, which makes online data mining critical. To cope with those challenges, embedded data mining [4] has been proposed to discover patterns and knowledge in embedded computing devices. Enegy-aware design has been a widely applied in embedded applications [5], where energy profiling or characterization is a key step to identify the energy bottleneck of the entire design and then to guide optimization accordingly. However, few studies have been proposed for the systematic energy characterization and optimization for embedded data mining.

978-1-4799-6177-1/14/$31.00 © 2014 IEEE

Among data mining algorithms, k-Nearest Neighbors (kNN) with Dynamic Time Warping (DTW) as the dissimilarity matric, namely the DTW-kNN framework, is one of the most widely used classification methods, which has been applied for speech recognition, financial market prediction, medical electronics application, scientific research and so on [6]. However, all these work do not take the energy optimization on embedded systems into consideration.

In this paper, we propose a systematic testbed for energy characterization for embedded data mining and apply the proposed testbed for optimizations of DTW-kNN. To the best of our knowledge, this is the first work to study the energy characterization and optimization of the data mining algorithm on embedded computing testbeds. And our contributions can be summarized as the following three aspects:

- We design an energy measurement testbed for data mining algorithms.
- We analyze the energy characterization of the DTW-kNN framework based on our proposed energy measurement testbed;
- Multiple optimization strategies are selected and implemented on the testbed to reduce total energy of the system.

The remainder of this paper is divided into 6 sections. Section II introduces the background of DTW, kNN and DTW-kNN framework. The detailed implementation of the energy measurement testbed is given in Section III. The energy characterization of the DTW-kNN framework is analyzed in Section IV. Based on Section IV, Section V introduces several energy optimization methods. In Section VI, the optimized DTW-kNN is configured to different experimental setups and the experimental results and analysis are also given. The paper is concluded in Section VII.

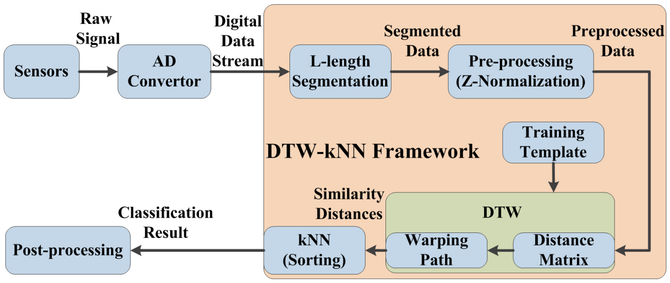


Fig. 1. Signal processing chain of embedded system in IOT. The DTW-kNN framework is integrated in the chain, including Segmentation, Pre-processing, DTW and kNN modules.

II. Background

A. The DTW-kNN Framework

The DTW-kNN framework is a widely applied classification framework. A typical signal processing chain of embedded system in IOT, integrating the DTW-kNN framework, is shown in Fig. 1. In general, the DTW-kNN framework consists of four functional modules, Segmentation, Pre-processing, DTW and kNN.

In the whole processing chain of IOT, the data captured from sensors is often regarded as raw analog data. Usually, raw data stream is converted into digital sequences with Analog-to-Digital (AD) convectors. Then digital data streams are divided into sequences with the fixed length in segmentation module. Pre-processing aims to make data more reliable and convenient to process, in which normalization is a common choice. And *Z-Normalization* (ZN) [7], is adopted in this paper. After Pre-pocessing, the DTW module utilizes *distance matrix calculation* (DMC) and *warping path calculation* (WPC) to present the similarity distance between the pre-processed sequence and template sequences. With the similarity distances, classification of the preprocessed sequence is determined by kNN module, which is mainly a *sorting* (SN) task. At last, The post-processing module processes the classification result with some high level algorithms.

B. Dynamic Time Warping

Dynamic Time Warping is a distance metric of similarity. There are two time sequences in Fig. 2(a), a sequence C of length n, as a candidate, and a sequence T of length m, where:

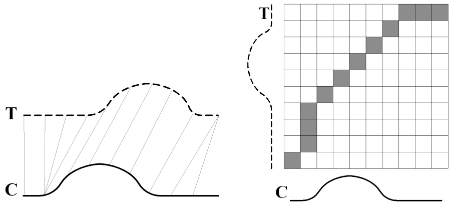
$$C = c_1, c_2, \cdots, c_i, \cdots, c_n, \tag{1}$$

$$T = t_1, t_2, \cdots, t_i, \cdots, t_m. \tag{2}$$

To measure the similarity of these two time sequence, DTW creates an n-by-m matrix M, which is called *distance matrix calculation*. The value of the (i^{th}, j^{th}) element in the matrix M represents the distance $d(c_i, t_j)$ between points c_i and t_j, as shown below:

$$M(i,j) = d(c_i, t_j) = (c_i - t_j)^2. \tag{3}$$

There are three well-known constrains in DTW [8]: *Boundary Conditions*, *Continuity Condition* and *Monotonic Condition*. *Boundary Conditions* means that the first/last point of C must correspond to the first/last point of T. *Continuity Condition* means that each element of the warping path in the matrix M must have two elements of the warping path around it except the first and the last points. *Monotonic Condition* requires that the extending direction of the warping path is right, or top or top-right.



(a) DTW matching (b) DTW warping path

Fig. 2. (a) DTW matching: T and C are two sequences, the lines indicates the matching relationship; (b) DTW warping path: the gray squares form the warping path between T and C on the distance matrix.

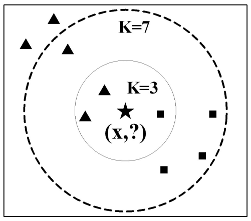


Fig. 3. kNN Rule: X is classified by a majority vote of k neighbors. The star in the center is classified as triangle when $k = 3$, while if $k = 7$, the star will be classified as square.

The shortest warping path through the matrix is derived, using [8]:

$$CD(i,j) = d(c_i, d_j) + min \begin{cases} CD(i, j-1) \\ CD(i-1, j) \\ CD(i-1, j-1) \end{cases}, \tag{4}$$

where $CD(i,j)$ is the current minimum cumulative distance. This procedure is called *warping path calculation*. The minimized cumulative distance found as showed in Fig. 2(b). And the final DTW distance is calculated as below:

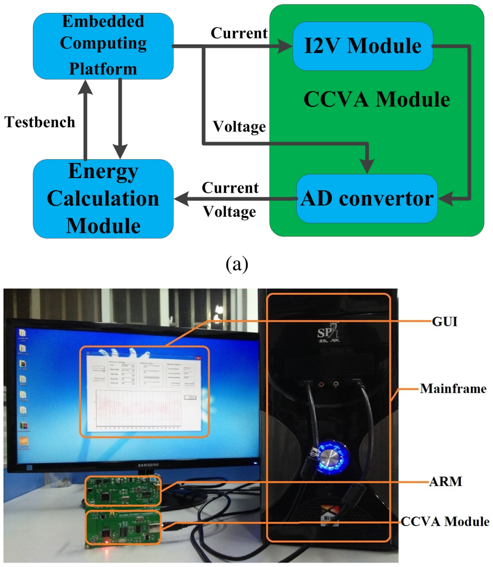
$$dtw = \sqrt{CD(n,m)}. \tag{5}$$

C. k-Nearest Neighbors

The k-Nearest Neighbors(kNN) algorithm is one of the well-investigated methods for pattern classification, which is used to determine the class of an unclassified point by the class of the k nearest points of it [9]. And k varies for different applications. A simple example of kNN is shown in Fig. 3.

III. Energy Measurement Testbed

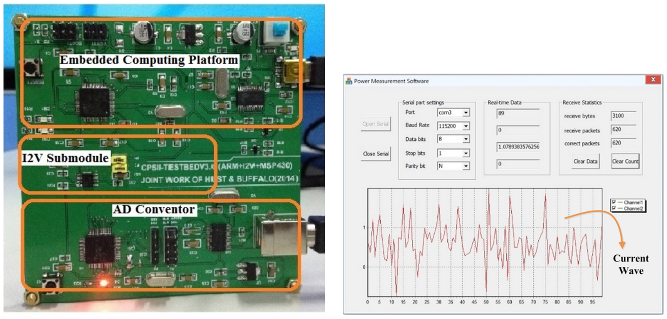
In this section our energy measurement testbed is presented. The proposed energy measurement testbed can quantitatively measure the dynamic energy consumption when data mining algorithms are running on the testbed.



(a)

(b)

Fig. 4. (a) The block diagram of energy measurement testbed; (b) the picture of the implementation of energy measurement testbed



Power Measurement Software

Serial port settings
Port: com3
Baud Rate: 115200
Data bits: 8
Stop bits: 1
Parity bit: N

Real-time Data
89
0
1.0789383576256
0

Receive Statistics
receive bytes: 3100
receive packets: 620
correct packets: 620

(a)

(b)

Fig. 5. (a) The PCB of the measurement testbed, which include the ARM module, the current-to-voltage module and the MSP430 module and (b) the GUI of the PC software

A. Energy Measurement Testbed Overview

The block diagram of the energy measurement testbed is shown in Fig. 4(a). The testbed consists of an embedded computing platform, a Current Conversion and Voltage Acquisition (CCVA) module, and an energy calculation module. The embedded computing platform provides a powerful platform to run data mining algorithms. The CCVA module converts the voltage and current signals from the embedded computing system to digital data and send it to the energy calculation module. Then the energy calculation module receives the current and voltage data to monitor the energy consumption of embedded computing platform in real-time. The picture of the testbed is shown in Fig. 4(b).

B. Embedded Computing Platform

The embedded computing platform can be implemented in various hardware components. And in this paper it consists of an ARM-based chip, a RS232-USB module, and a JTAG module. In order to meet the requirements of high performance and low energy consumption, STM32F103 is chosen, which is based on ARM Cortex-M3. Every chip has reserved at least one measurement probe to measure its electric voltage and current. These probes for current are connected to the current-to-voltage module.

DTW-kNN framework is implemented on the embedded computing platform. In order to measure energy consumptions of different stages of programs, application phase tags are added in programs, which send changes to MSP430. Different tags mark different operating stages.

C. Current Conversion and Voltage Acquisition Module

The Current Conversion and Voltage Acquisition module includes two submodules, I2V and AD convertor. The I2V converts measured currents to voltages which are then processed by AD converter. The I2V submodule consists of a current sensing chip. Based on considerations of accuracy and measuring range, we choose precision, high-side current-sense amplifier MAX471 to measure currents. The MAX471 is a complete, bidirectional, high-side current-sense amplifier.

The AD convertor is implemented in a MSP430 microcontroller and it has two responsibilities: converting voltages into digital data with a 12-bit AD converter and transfering the digital data and application phase tags to the PC software. The voltages include the voltage from the probes in embedded computing platform and the I2V submodule.

D. Energy Calculation Module

The energy calculation module is implemented in a Personal Computer (PC). The PC software reads and saves the measured data and the application phase tags. Data processing and analysis are also presented. The real-time curves and results are displayed with a graphical user interface (GUI), as shown in Fig. 5(b).

IV. Energy Characterization

With the developped energy measurement testbed, we design and perform detailed experiments to characterize the energy consumption of the DTW-kNN framework.

A. Characterization Experiment

In the experiments, datasets containing segmented data from a popular data warehouse for classification and clustering [6] are used, which omits data stream acquisition, AD conventor and L-length segmentation. As the RAM and ROM are limited in the ARM chip, 5 datasets with short sequence length are selected. Detailed information of the selected 5 datasets in the experiments is shown in TABLE I. Three k values, 1, 3 and 5, are selected. The CPU clock speed of the ARM chip in the energy measurement testbed is set to 72MHz. The sizes of training set and test set of the 5 datasets are 10 and 100, respectively.

Distance matrix calculation and *warping path calculation* in DTW require a $2 \times N \times N$ memory space, which may exceed the capacity of RAM of the ARM chip when the sequence length n is large. Based on the idea that the minimum coupling part of distance matrix and warping path matrix is just one row, we propose a memory-efficient operation method to allow

TABLE I. Detailed information of the selected 5 datasets [6]

NO.	Name	Number of classes	Segment Length	Signal Range
1	Medical Images (MI)	10	99	[-2.0,6.6]
2	ECG	2	96	[-3.0, 4.1]
3	Mote Strain (MS)	2	84	[-7.8,3.4]
4	SonyAIBORobot Surface (SS)	2	70	[-3.5,3.7]
5	Italy Energy Demand (IPD)	2	24	[-2.4,3.3]

TABLE II. Energy characterization of DTW-kNN framework

Datasets	k	Total Energy (mJ)	Energy percentages of different parts			
			ZN	DMC	WPC	SN
Medical Images (MI)	1	15780	3.06%	89.34%	7.44%	0.15%
ECG	1	14246	2.44%	89.89%	7.49%	0.18%
Mote Strain (MS)	1	11336	2.94%	89.42%	7.45%	0.18%
SonyAIBORobot Surface (SS)	1	7804	2.83%	89.49%	7.45%	0.22%
Italy Energy Demand (IPD)	1	906	2.39%	89.50%	7.46%	0.66%
Medical Images (MI)	3	15821	3.05%	89.11%	7.43%	0.42%
ECG	3	14290	2.44%	89.62%	7.47%	0.48%
Mote Strain (MS)	3	11372	2.93%	89.14%	7.43%	0.50%
SonyAIBORobot Surface (SS)	3	7834	2.82%	89.15%	7.43%	0.59%
Italy Energy Demand (IPD)	3	916	2.36%	88.51%	7.38%	1.75%
Medical Images (MI)	5	15853	3.04%	88.93%	7.41%	0.62%
ECG	5	14323	2.43%	89.4%	7.45%	0.71%
Mote Strain (MS)	5	11399	2.93%	88.93%	7.41%	0.73%
SonyAIBORobot Surface (SS)	5	7856	2.81%	88.90%	7.41%	0.88%
Italy Energy Demand (IPD)	5	924	2.34%	87.77%	7.31%	2.58%

all the datasets run on the ARM-based testbed. Our method calculates a row of the distance matrix first, and then calculate a row of the warping path matrix. So only $2 \times N$ storage is needed. This process will iterate until the calculation of warping path matrix ends. Therefore it cuts down the memory occupation with no impact on arithmetic calculation.

B. Characterization Analysis

The results of the characterization experiments are shown in TABLE II. The total energy consumption is larger as the sequence length increases under the same k setup. For the same dataset, total energy grows about 1% as 2 unit increment of k . The energy percentages of different parts of the five datasets under different k value setups are almost the same. The most energy-expensive step is *distance matrix calculation*, which accounts for 89.14% on average of the total energy consumption. While *warping path calculation* and *normalization* accounts for around 7% and 2%. And *sorting* is the least energy consumer, accounting for around 1%. The *distance matrix calculation* is made up of multiply operations and its asymptotic time complexity is $O(n^2)$, so it is rather energy-consuming. While the *warping path calculation* is full of logical operations with an asymptotic time complexity of $O(n^2)$. As logical operations are much less energy-expensive than multiply operations, the energy consumption of *warping path calculation* is much less than that of *distance matrix calculation*. The asymptotic time complexity of both *normalization* and *sorting* are $O(n)$, so their energy percentages are very small. As multiply and division operations in *normalization* are more energy-expensive than logical operations in *sorting*, the percentage of *sorting* is much smaller.

The DTW calculation includes two most energy-consuming submodules, *distance matrix calculation* and *warping path calculation*, which totally account for as much as 97% of the total energy. From TABLE II we can see that DTW calculation is the bottleneck of the whole DTW-kNN framework.

V. Strategies of Energy Optimization

With the energy characterization of the DTW-kNN framework observed in Section IV, we will employ and discuss several optimization methods in this section. And we firmly claim that all the selected and proposed methods have no influence on accuracy.

A. The Squared Distance

The DTW distances need to compute square root. In fact, this computation makes no difference of the rankings in kNN [9]. Compared with Formula (5), the optimized version of Formula (5) is shown below.

$$dtw = CD(n, m). \tag{6}$$

Removing square root computing will remove square root operations, which will potentially save time and energy.

B. Early Abandon of DTW

During the calculation of DTW in the DTW-1NN framework, early abandon of DTW can be achieved. This is due to the fact that the warping path obeys the rule of *Continuity Condition* and there exist at least 1 element in a row that belongs to the warping path. Suppose that an exact DTW distance dtw_1 is calculated. During the next calculation of DTW, if the minimum $\sqrt{CD(i,j)}$ of the current row exceeds dtw_1, the final dtw of the current DTW must exceed dtw_1 and the calculation of it can be abandoned early. The mechanism of other conditions of k in DTW-kNN framework is the same.

C. Lower Bound and Indexing DTW

Lower bounding functions are used to estimate the lower bound of DTW distances. Several lower bound functions are selected for indexing DTW. The lower bounding function proposed by Kim [10] (referred LB_Kim hereafter), by Yi [11] (referred LB_Yi hereafter), and a revised version of LB_Kim by Rakthanmanon [8](referred LB_Kim2 hereafter) are selected.

Indexing with the above lower bounding functions, a kNN Indexing algorithm is introduced in Algorithm **??**. It is a modification of the algorithm used for indexing time series in [12].

As shown in Algorithm 1, Q stands for query sequence, while T stands for training sequences. K is the desired number of numbers in kNN. Two min priority queue, queueLB, queueResult , are used. queueLB is sorted by the lower bound, while the queueResult is sorted by the DTW distance. Firstly, all lower bounds between Q and T are pushed into queueLB. Then the algorithm pops out the first element of queueLB in each cycle until queueLB is empty. During each cycle, if the length of queueResult is less than K, the exact DTW distance of first element is calculated and pushed into queueResult. If the LBdistance of first element is equal to or greater than the DTW distance of the last element in queueResult, the algorithm returns. If, on the other hand, the DTW distance of the first element is less than the DTW distance of the last element in queueResult, first element will replace the last element in queueResult.

Algorithm 1 kNNIndexing(Q,K)

```
Variable: queueLB, queueResult:MinPriorityQueue;
queueLB.push($T$, LBdistance($Q$,$T$));
while not queueLB.IsEmpty() do
    first=queueLB.first();
    if queueResult.length()$<K$ then:
        queueResult.push(first, DTWdistance(first,$Q$));
    else if first.LBdistance$>=$
                queueResult.last().DTWdistance then
        return queueResult;
    else if DTWdistance(first,$Q$)$<$
                queueResult.last().DTWdistance then
        queueResult.popLast();
        queueResult.push(first, DTWdistance(first,$Q$));
    else
        continue;
    end if
end while
return queueResult;
```

TABLE III. Detailed information of the Voltage and I_{static}

Voltage	I_{static} (mA) with CPU clock rates							
(V)	16MHz	24MHz	32MHz	40MHz	48MHz	56MHz	64MHz	72MHz
3.325	4.28	6.25	8.31	10.30	12.32	14.39	16.71	18.45

VI. Evaluation

In this section, we carry out a comprehensive set of experiments to evaluate the DTW-kNN framework with selected optimization strategies.

A. Experimental Setup

The same 5 datasets are selected as shown in TABLE I. We also use the memory operation method of DTW as Section IV. The energy characterization of the DTW-kNN framework in Section IV shows that k does not have significant influence on the energy characterization. So in this part DTW-1NN is adopted for simplification. The sizes of training set and test set of the 5 datasets are 10 and 100, respectively. The energy consumption of the ARM-based testbed is calculated as shown below.

$$e = t \times V \times (I_{knn-dtw} - I_{static}), \tag{7}$$

where e stands for energy consumption, and t is the running time of the program. V is the voltage. $I_{knn-dtw}$ is the average current when running programs. I_{static} is the average current of the program doing nothing. The detail information is shown in TABLE III. And CPU clock rate is set to 72MHz by default.

B. Experimental Result and Analysis

1) The Squared Distance: The comparison between the squared distance and the squared root distance is tested. TABLE IV shows the results of the energy comparison of squared distance and squared root distance. We can see that the energy of squared distance is less than that of squared root distance and the squared distance can reduce energy consumption with about 1%. The difference between the two strategies is not so significant on all 5 datasets. This is because the square root operation is executed only once of every DTW distance calculating. Since the computing complexity of DTW is $O(n^2)$, the energy of calculating square root becomes a subtle factor. Also, when comparing the energy reduction between different datasets, we can find that datasets with long sequence gain more energy reduction, while short-sequence datasets have less reduction, such as IPD has the lowest reduction. This is because that long sequence is probable to have a relatively big DTW distance.

TABLE IV. Energy comparison of squared distance and squared root distance

Datasets	DTW-kNN energy consumption (mJ)				
	MI	ECG	MS	SS	IPD
Squared Root Distance	15779	14246	11336	7804	905.81
Squared Distance	15323	14040	11004	7722	898.56
Reduction	2.9%	1.4%	2.9%	1.0%	0.8%

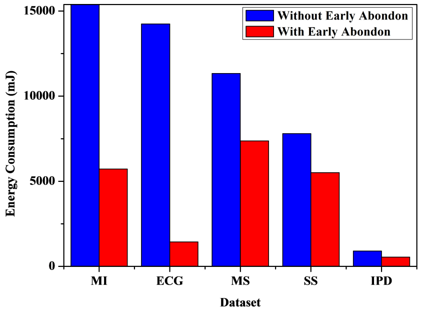


Fig. 6. Energy comparison of early abandon optimization and no optimization

2) Early Abandon of DTW: Early abandon of DTW is implemented without no other optimizations. As shown in Fig. 6, the energy reduction of early abandon optimization method varies from 29.5% to 89.9%. This is due to the fact that it is random to meet the best matching sequence for the test sequence. If the best matching sequence is calculated in the first step, the other calculation can be abandoned very early and the energy reduction can be very huge. Unfortunately, if the calculations flow of DTW is from the largest DTW distance to the least DTW distance in a sequential chain, the reduction can be rather low.

3) Lower Bound and Indexing DTW: TABEL V shows the energy consumption varies with different lower bounding methods and datasets. The energy reductions vary on different datasets. But the three lower bounding methods have not significant difference. Compared with the DTW-kNN framework without optimization in TABLE II, the average energy reductions are around 50%.

4) Put it all together: In this part all the optimization strategies are implemented and tested together.

TABLE VI shows that with full optimization, the energy reduction varies with different datasets. The highest reduction

TABLE V. Energy consumption of different lower bounding methods

LB method	Datasets				
	MI	ECG	MS	SS	IPD
LB_KIM	74.9%	58.5%	51.6%	33.5%	47.4%
LB_KIM2	73.7%	57.5%	45.3%	35.5%	49.4%
LB_Yi	74.6%	56.4%	52.8%	35.0%	43.5%
Average	74.4%	57.5%	49.9%	34.7%	46.8%

TABLE VI. ENERGY REDUCTION OF FULL OPTIMIZATION FOR DTW-KNN FRAMEWORK WITH A 72MHZ CPU CLOCK RATE

Full Optimization	Datasets				
	MI	ECG	MS	SS	IPD
SD+EA+LB_KIM	74.6%	59.4%	52.5%	34.8%	46.9%
SD+EA+LB_KIM2	73.3%	56.2%	44.9%	35.1%	50.1%
SD+EA+LB_Yi	74.3%	56.9%	50.0%	34.6%	44.5%
Average	74.1%	57.5%	49.1%	34.8%	47.2%

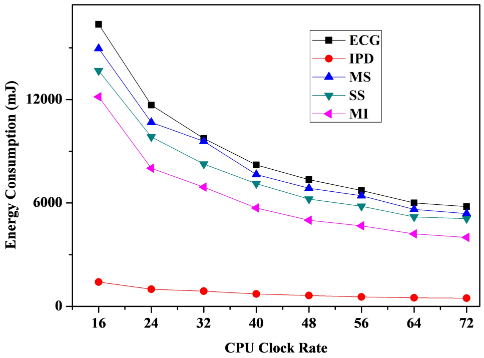


Fig. 7. Frequency scaling on dynamic energy

is as much as 74.6%. The energy reduction of the full optimization is not a simple addition of energy reduction of each optimization method. This is caused by the fact that all the optimization methods are coupled by the DTW-kNN framework.

We can take a close look at the comparison of TABLE V and TABLE VI. All the energy reduction seems very similar, with difference no more than 2%. It demonstrates that when we put all the optimization strategies together, lower bound and indexing DTW method dominates the whole optimization. For the squared distance method, the energy comsumption of square root operation only takes up a very small proportion, so it does not have a big impact. While for early abandon strategy, its performance depends on the position of the best match with great randomness. However, lower bounding exploits the signal's intrinsic feature better and indexing can remove the randomness. Therefore, lower bounding and indexing DTW strategy plays a dominating role in the combination of all optimization strategies.

C. Frequency Scaling on Dynamic Energy

In this part all the optimization are implemented and tested with different CPU clock rate. For simplification, the lower bounding method LB_KIM is selected.The result is illustrated in Fig. 7. From Fig. 7, all the energy consumption of 5 datasets decreases as the CPU clock rate increases. Except IPD dataset, other datasets's energy consumption all decrease very fast as CPU clock increases, while IPD suffers from a very slow decreasing rate. This is related to the sequence length in the dataset. The energy consumption increases by quadratic speed of segment length in the dataset. We can also observe that the absolute energy consumption of other four dataset is not proportional to their segment length. This is involved with our optimization strategy and intrinsic property of signal. In sum, it clearly indicates that a higher CPU clock rate will save time and energy at the same time.

VII. CONCLUSION AND FUTURE WORK

Within data mining, the DTW-kNN framework is a widely used strategy in many areas. With more and more emphasis on energy of internet of things (IOT) and mobile internet, energy characterization of data mining method including DTW-kNN is very critical. In this paper, the energy characterization and optimization of DTW-kNN framework are analyzed with a comprehensive set of experiments. And Several meaningful results can be made:

- The bottleneck of the DTW-kNN framework is *distance matrix calculation* accounting for 89.14% on average of the total energy consumption.
- The energy reduction of squared distance, early abandon and lower bounding methods are about 1%, from 29.5% to 89.9% and about 50% respectively.
- When all optimization methods are implemented, the energy reduction can be as much as 74.6%.

In the future work, more optimization methods involved with tradeoffs of speed, energy and accuracy will be discussed. Also a benchmark work of energy characterization of distance measures and classification methods will be presented.

REFERENCES

[1] T.-c. Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 164–181, 2011.

[2] G. Cormode, "Fundamentals of analyzing and mining data streams," in *Tutorial at Workshop on Data Stream Analysis, Caserta, Italy*, 2007.

[3] J. Li, X. Xu, H. Zhao, Y. Hu, and T. Z. Qiu, "An energy-efficient sub-nyquist sampling method based on compressed sensing in wireless sensor network for vehicle detection," in *Connected Vehicles and Expo (ICCVE), 2013 International Conference on*. IEEE, 2013, pp. 314–319.

[4] R. U. Fayyad, Usama, "Evolving data into mining solutions for insights," *Communications of the ACM*, vol. 45, no. 8, pp. 28–31, 2002.

[5] L. B. Simunic, Tajana and G. D. Micheli, "Energy-efficient design of battery-powered embedded systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 1, pp. 15–28, 2001.

[6] E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana, "The ucr time series classification/clustering homepage," *URL= http://www. cs. ucr. edu/˜ eamonn/time_series_data*, 2006.

[7] E. Keogh and S. Kasetty, "On the need for time series data mining benchmarks: a survey and empirical demonstration," *Data Mining and knowledge discovery*, vol. 7, no. 4, pp. 349–371, 2003.

[8] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 262–270.

[9] J. Tarango, E. Keogh, and P. Brisk, "Instruction set extensions for dynamic time warping," in *Proceedings of the Ninth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*. IEEE Press, 2013, p. 18.

[10] S.-W. Kim, S. Park, and W. W. Chu, "An index-based approach for similarity search supporting time warping in large sequence databases," in *Data Engineering, 2001. Proceedings. 17th International Conference on*. IEEE, 2001, pp. 607–614.

[11] B.-K. Yi, H. Jagadish, and C. Faloutsos, "Efficient retrieval of similar time sequences under time warping," in *Data Engineering, 1998. Proceedings., 14th International Conference on*. IEEE, 1998, pp. 201–208.

[12] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Locally adaptive dimensionality reduction for indexing large time series databases," *ACM SIGMOD Record*, vol. 30, no. 2, pp. 151–162, 2001.