International Journal of Web Services Research
Volume 14 • Issue 2 • April-June 2017

A Fast Shapelet Discovery Algorithm Based on Important Data Points

Cun Ji, School of Computer Science and Technology, Shandong University, Jinan, China

Chao Zhao, School of Computer Science and Technology, Shandong University, Jinan, China

Li Pan, School of Computer Science and Technology, Shandong University, Jinan, China & Engineering Research Center of Digital Media Technology, Ministry of Education, Jinan, China

Shijun Liu, School of Computer Science and Technology, Shandong University, Jinan, China & Engineering Research Center of Digital Media Technology, Ministry of Education, Jinan, China

Chenglei Yang, School of Computer Science and Technology, Shandong University, Jinan, China & Engineering Research Center of Digital Media Technology, Ministry of Education, Jinan, China

Lei Wu, School of Computer Science and Technology, Shandong University, Jinan, China & Engineering Research Center of Digital Media Technology, Ministry of Education, Jinan, China

ABSTRACT

Time series classification (TSC) has attracted significant interest over the past decade. A shapelet is one fragment of a time series that can represent class characteristics of the time series. A classifier based on shapelets is interpretable, more accurate, and faster. However, the time it takes to find shapelets is enormous. This article will propose a fast shapelet (FS) discovery algorithm based on important data points (IDPs). First, the algorithm will identify IDPs. Next, the subsequence containing one or more IDPs will be selected as a candidate shapelet. Finally, the best shapelets will be selected. Results will show that the proposed algorithm reduces the shapelet discovery time by approximately 14.0% while maintaining the same level of classification accuracy rates.

KEYWORDS

Classification, *Data Mining*, *Important Data Points*, *Internet of Things*, *Shapelet*, *Time Series Data*

1. INTRODUCTION

The internet of things (IoT) is made up of small sensors and actuators embedded in objects with internet access. It plays a key role in solving challenges faced in today's society (Nunes et al., 2016; Sampaio, Lima, Mendonça, & Filho, 2013). Sensors in an IoT system collect observation data with equal intervals. Data collected in an IoT system is time series data.

Time series utilizes data points indexed in time order (either listed or graphed). This collection of values is obtained over time using sequential measurements. It is characterized by its numerical and continuous nature (Esling & Agon, 2012; Fu, 2011).

Time series is always considered as a whole rather than an individual numerical field. The high dimensionality, high feature correlation, and typically high levels of noise provide an interesting research problem (Gabr & Fatehy, 2013; Keogh & Kasetty, 2003; Ye & Keogh, 2009). Effective TSC has been an important research problem for both academic researchers and industry practitioners.

In TSC, an unlabeled time series is assigned to one of at least two predefined classes (Keogh & Kasetty, 2003). TSC arises in many real-world fields, including: electrocardiogram classification; fault detection and identification of physical systems; automotive preventive diagnosis; gesture recognition;

DOI: 10.4018/IJWSR.2017040104

alarm interpretation of telecommunication networks; data sensor analysis; speaker identification and/or authentication; aerospace health monitoring, etc. (Prieto, Alonso-González, & Rodríguez, 2015).

Many classification algorithms can be applied to time series (for example, classification trees, nearest neighbors, discriminant analysis, and iterative classification). Empirical evidence strongly suggests that classification based on time series shapelets will outperform many classification algorithms (He, Dong, Zhuang, Shang, & Shi, 2012). Shapelets are discriminative subsequences which have the property that the minimum distance between a shapelet and the time series is a good predictor for TSC (Wistuba, Grabocka, & Schmidt-Thieme, 2015). Algorithms based on shapelets are interpretable, more accurate, and faster than state-of-the-art classifiers (Mueen, Keogh, & Young, 2011; Ye & Keogh, 2009, 2011).

A shapelet is a time series subsequence representative of class membership. Algorithms based on shapelets are interpretable, more accurate, and faster than state-of-the-art classifiers. The time complexity of the shapelet selection process is high although shapelets are computed offline (Rakthanmanon & Keogh, 2013).

For this, a FS discovery algorithm based on IDPs (FS-IDPs) is proposed. First, the algorithm identifies IDPs. Next, only subsequences containing one or more IDPs are selected as candidate shapelets. Finally, the best shapelets are selected from the candidates. Through IDPs, the number of shapelet candidates is reduced. This leads to the time-saving shapelet discovery.

In this article, the authors will propose a FS-IDPs. The algorithm will use IDPs to speed up the shapelet discovery process. Next, comparison experiments among different shapelet discovery algorithms will be conducted. Experiment results will show that the algorithm will speed up the shapelet discovery process while maintaining the same level of classification accuracy rates. Then, the article will introduce definitions, summarize related work, and present the proposed FS-IDPs algorithm. Finally, the article will describe experiments, show results, and present a conclusion.

2. DEFINITION

Definition 1: A *time series* (*T*) is an ordered list of real-valued variables: $T = t_1, t_2 \dots t_m$. Typically, data points $t_1, t_2 \dots t_m$ are arranged by temporal order and spaced at equal time intervals. The length of time series *T* is *m*. The length of time series is also noted as $|T|$.

Definition 2: A *time series subsequence (S)* is a contiguous sequence of a time series. Subsequence *S* of length *l* of time series *T* starting at position *i* can be written as $S = T_i^l = t_i, t_{i+1} \dots t_{i-l+1}$.

Definition 3: A *dataset* (*D*) is a set of time series: $D = \{T_1, T_2 \dots T_{|D|}\}$. $|D|$ means the number of time series in dataset *D*. Note that the lengths for each time series may not be equal.

Definition 4: The *distance between time series (dist(T, R))* is a distance function that takes two time series $T = t_1, t_2 \dots t_m$ and $R = r_1, r_2 \dots r_m$ (of the same length as inputs) and returns a non-negative value.

This article will use the Euclidean distance. This is calculated as shown in equation (1). It is also applicable to subsequences of the same length.

$$dist\left(T,R\right) = \sqrt[2]{\sum_{i=1}^{m}\left(t_i - r_i\right)^2} \tag{1}$$

Definition 5: The *distance from the time series to the subsequence (subDist(T, S))* is a distance function that takes time series *T* and subsequence *S* as inputs and returns a non-negative value. This is the minimum possible distance from *T* to *S*. It is calculated as in equation (2).

$$subDist\left(T,S\right) = min\left(dist\left(T_1^l,S\right),\ldots dist\left(T_{m-l+1}^l,S\right)\right) \tag{2}$$

In equation (2), *m* is the length of time series *T* and *l* is the length of subsequence *S*.

Definition 6: *Entropy (e)*: Suppose that dataset *D* contains time series from *c* different classes. The probability of time series in class *i* is p_i. The entropy of *D* can be calculated as in equation (3).

$$e\left(D\right) = -\sum_{i=1}^{c} p_i \log p_i \tag{3}$$

Definition 7: A *split (sp)* is a two-tuples <*s, d*> of a subsequence *s* and distance threshold *d*. This can separate the dataset into two smaller datasets, D_L and D_R. The time series in *D* which distance to *s* is bigger than *d* is put into D_R, otherwise it is put into D_L. The number of time series in D_L and D_R are n_L and n_R, respectively.

Definition 8: The *information gain (gain)* of a split *sp* can be calculated as in equation (4).

$$gain\left(sp\right) = e\left(D\right) - \frac{n_L}{n} e\left(D_L\right) - \frac{n_R}{n} e\left(D_R\right) \tag{4}$$

Definition 9: A *separation gap (gap)* is the distance between two different sides of the given split *sp*. It can be calculated as in equation (5).

$$gap\left(sp\right) = \left| \frac{1}{n_L} \sum_{t_L \in D_L} dist\left(t_L, s\right) - \frac{1}{n_R} \sum_{t_R \in D_R} dist\left(t_R, s\right) \right| \tag{5}$$

Definition 10: A *shapelet* is a split that separates the dataset into two smaller datasets with the maximum information gain. Ties are broken by maximizing the separation gap.

A more complete definition of shapelet can be found using the following resources: He et al. (2012), Mueen et al. (2011), and Ye and Keogh (2009).

3. RELATED WORK

Since its introduction in 2009 (Ye & Keogh, 2009), shapelet has concerned many researchers. First, a shapelet classifier will classify new instance faster because it is more compact than many alternatives. Second, shapelets are straightly interpretable. Third, shapelets allow for the detection of shape-based similarity of subsequences. This type of similarity can be hard to detect with algorithms based on whole series.

Many shapelet-based algorithms have been proposed. These algorithms can be divided into three categories:

1. **Shapelet Discovery Algorithms** (Mueen et al., 2011; Rakthanmanon & Keogh, 2013; Ye & Keogh, 2009). These algorithms embed the shapelet-discovery algorithm into a decision tree, finding a new shapelet at each node of the tree (Hills, Lines, Baranauskas, Mapp, & Bagnall, 2014).
2. **Shapelet Transform Algorithms** (Hills et al., 2014; Lines, Davis, Hills, & Bagnall, 2012). These algorithms optimize the process of shapelet selection and adopt various classification strategies. Through transforming by shapelets, TSC problems can be viewed as general classification problems.
3. **Learning Shapelets Algorithms** (Grabocka, Schilling, Wistuba, & Schmidt-Thieme, 2014; Hou, Kwok, & Zurada, 2016; Shah, Grabocka, Schilling, Wistuba, & Schmidt-Thieme, 2016). These methods adopt a heuristic gradient descent shapelet search procedure rather than enumeration (Bagnall, Bostrom, Large, & Lines, 2016). They directly learn the shapelets jointly with the classifier.

Shapelet discovery algorithms form the foundation for the other algorithms. Shapelet discovery algorithms have also adopted the most acceleration technologies. Currently, shapelet discovery algorithms are the fastest of the three categories.

The brute force shapelet discovery algorithm was introduced by Ye and Keogh (2009). This algorithm generates all possible candidates. After testing the candidates, the algorithm returns the best option. The obvious weakness of the brute force shapelet discovery algorithm is the very slow training time. The time complexity of the brute force algorithm is up to O(n^2m^4), with *n* being the number of time series in dataset and *m* being the length of time series. Many methods have been proposed to speed up the shapelet discovery algorithm.

Ye and Keogh (2011) developed two speed-up methods: (1) subsequence distance early abandon (SDEA) and (2) admissible entropy pruning (AEP). In SDEA, the computation is abandoned once the distance is larger than the current smallest distance. SDEA can reduce the runtime by a factor of two. AEP calculates a cheap-to-compute upper bound of the information gain. It uses this to admissibly prune certain candidates. AEP reduces the runtime by more than two orders of magnitude.

The current state-of-the-art algorithm guaranteed to find the same shapelet with the brute force algorithm (Mueen et al., 2011). They accelerated the time required to find the same shapelet by reusing computations and pruning the search space. Their algorithm used a matrix to cache the distance computations for future use. It then applied the triangle inequality to prune candidates. This resulted in a time complexity of $O(n^2m^3)$ and a memory footprint of up to $O(nm^2)$. Note that the speed-up method can only handle time series without normalization.

Improvements in Ye and Keogh (2011) and Mueen et al., (2011) guaranteed to find the same shapelet with the brute force algorithm. Some speed-up methods which not guaranteed to find the same shapelet with the brute force algorithm are also put forward.

Chang, Deka, Hwu, and Roth (2012) introduced an implementation on highly parallel graphics process units (GPUs). Through hardware-based optimization, they significantly reduced the running time of the shapelet discovery algorithm. However, the cost of hardware optimization was expensive.

He et al. (2012) reduced the running time by elaborating on the usage of infrequent shapelet candidates. They supposed that discriminative subsequences were infrequent compared to other subsequences. This assumption may not be tenable in specific databases.

Until now, the time complexity of the fastest shapelet discovery algorithm was $O(nm^2)$. Rakthanmanon and Keogh (2013) proposed FS. Exploiting projections on the symbolic aggregate approximation (SAX) representation was also used to find shapelet in FS. Zhang, Zhang, Wen, and Yuan (2016) accelerated FS with key points, generating shapelet candidates without repetition. By doing this, they sped up the discovery process. However, the accuracy rates of their method were somewhat lower than FS.

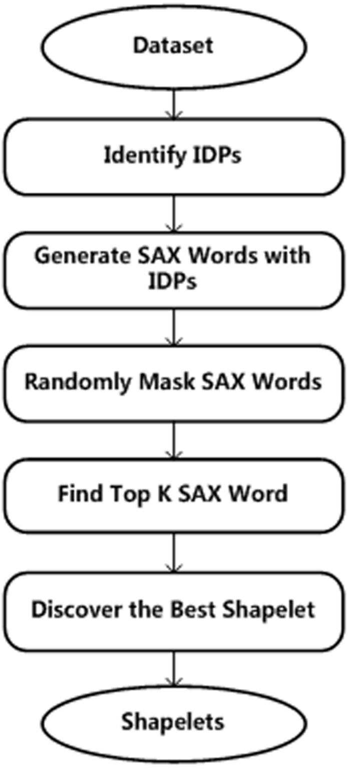
Previously, the authors used discrete Fourier transform (DFT) to filter noise and symbolic Fourier approximation to represent time series (Ji, Hu, Liu, Yang, Pan, & Wu, 2016). This can be called the FS-SFA algorithm. FS-SFA has a slightly higher classification accuracy rate than FS in sensor reading time series. However, FS-SFA uses multiple coefficient binning technique to quantization intervals by computed from the samples, which introduce extra time cost. Therefore, the time consume of FS-SFA is a bit higher than FS. It should be noted that all three of these methods have the same level of time complexity at $O(nm^2)$.

In this article, the authors expected to reduce training time while maintaining the same level of classification accuracy rates. In introducing IDPs to accelerate the shapelet discovery process, the article is a substantial extension of the authors' previous paper (Ji et al., 2016). First, IDPs are introduced to TSC. The new method, FS-IDPs, uses only subsequence containing one or more selected IDPs as shapelet candidates. Through IDPs, the number of shapelet candidates is reduced. This, in turn, speeds up the shapelet discovery procession. Second, the article includes more experiments to compare to other methods.

4. THE FAST SHAPELET DISCOVERY ALGORITHM BASED ON IMPORTANT DATA POINTS (FS-IDPS)

Figure 1 shows the process flow of algorithm FS-IDPs. There are five processes in FS-IDPs: (1) identify IDPs; (2) generate SAX words based on IDPs; (3) randomly mask SAX words; (4) find the top *k* SAX word; and (5) identify the best shapelet. These five processes will be described in the following subsection.

Figure 1. Process flow diagram of FS-IDPs



4.1. Identify IDPs

The speedup strategy used in FS-IDPs is aimed at reducing the number of shapelet candidates. However, the overall goal is the accuracy of the classification. As Zhang et al. (2016) described, a shapelet must have certain significance. They use subsequence contain key points as shapelet candidates. However, the accuracy rates of their method are a little lower than FS. In this article, a new method to reduce the number of shapelet candidates with the accuracy rates retaining the same level is presented by introducing the IDPs.

IDPs were introduced in 2016 to represent time series (Ji, Liu, Yang, Wu, Pan, & Meng, 2016). In this article, IDPs are defined as those points satisfying the following conditions:

1. The point is in the sequence with maximum weight.
2. The point is at the maximum distance from the line of best fit of the sequence.

In this definition, the sequence weight is related to the distances between the points and the fitting line of the sequence. The weight of one sequence is expressed as in equation (6):

$$weight = \max\left\{dist_{sum}, 2 * dist_{\max}\right\} \tag{6}$$

In equation (6), $dist_{sum}$ is the sum of the distances of all points in the sequence; $dist_{max}$ is the maximum among these distances. These distances are the fitting errors of the points to the fitting line of the sequence.

Ji, Liu, et al. (2016) identified IDPs by the fitting errors threshold. However, the fitting error is difficult to determine. In this article, IDPs are identified by a process until a given number has been reached. The process of identifying IDPs is shown in Figure 2.

As shown in Figure 2, the beginning and ending points are selected as IDPs in default. New IDPs are then selected and the information is repeatedly updated until the number of IDPs reaches the given number.

4.2. Generate SAX Word with IDPs

A shapelet is a subsequence that can discriminate time series from different classes. Therefore, the subsequence with variations in value is a more reasonable choice as a shapelet (Zhang et al., 2016). In the original application of IDPs, the subsequence of the two adjacent IDPs can be fitted by a straight line. Variations in value exist if one subsequence is completely located between two adjacent IDPs. For this reason, IDPs can be used in filtering shapelet candidates.

In Figure 3, IDPs are marked by red dots. As shown, the subsequences completely located between two adjacent IDPs are filtered out. For example, subsequence S1 (marked in green) is abandoned. The subsequences crossing one or more IDPs are used as shapelet candidates. For example, subsequence S2 (marked in red) is selected as shapelet candidates. If one subsequence is selected as shapelet candidates, it is represented by SAX (Lin, Keogh, Wei, & Lonardi, 2007; Wei, Keogh, & Xi, 2006) and carried through the follow-up processing.

4.3. Randomly Mask SAX Words

When using SAX to represent subsequence, a false dismissal can occur when there are two time series differing by a tiny error and producing two SAX words (this holds true for any discretization method, Rakthanmanon & Keogh, 2013). For an example of false dismissals of SAX, see Figure 4. As shown in Figure 4, the SAX represents two similar, yet different, subsequences ("abccdd" and "abbcdd"). The solution to false dismissals is to randomly mask SAX words.

Figure 2. Process of identifying IDPs

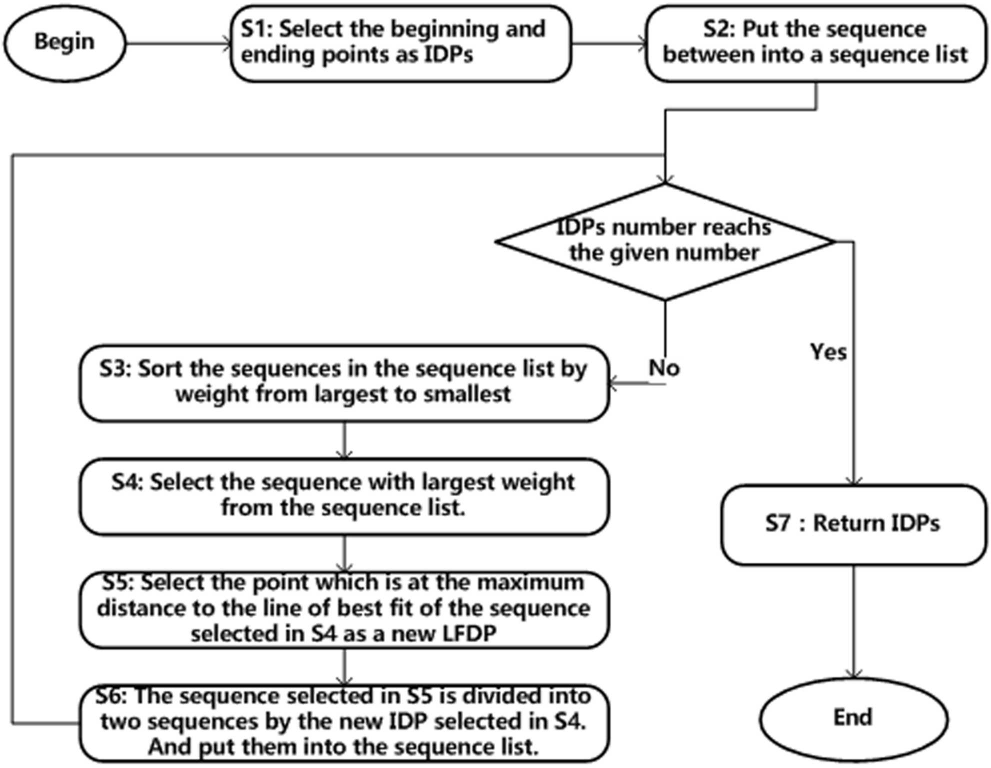
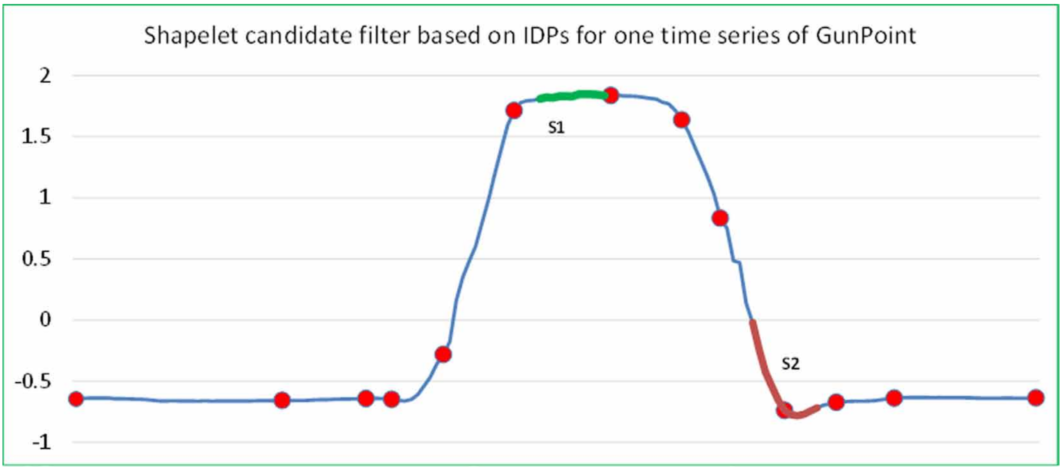
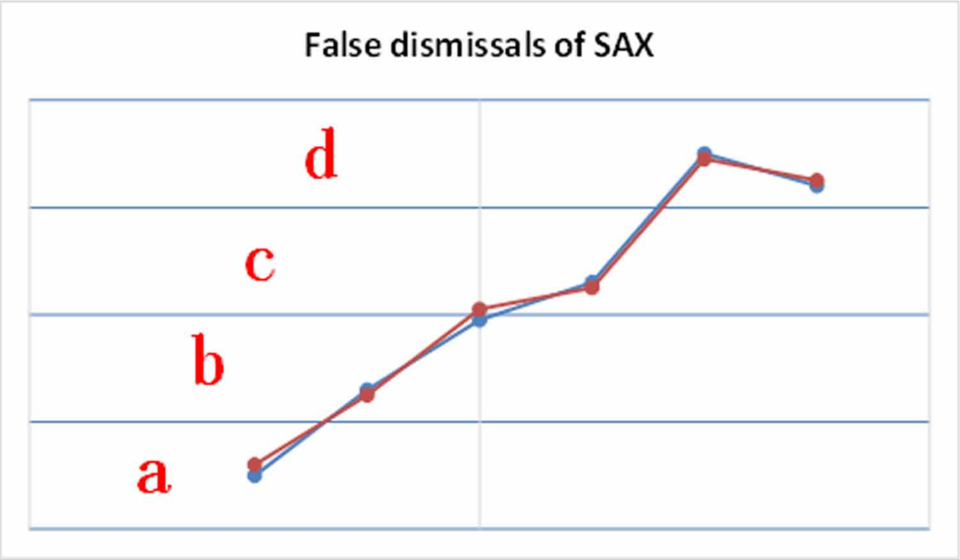

Begin
S1: Select the beginning and ending points as IDPs
S2: Put the sequence between into a sequence list
IDPs number reachs the given number
No
Yes
S3: Sort the sequences in the sequence list by weight from largest to smallest
S4: Select the sequence with largest weight from the sequence list.
S5: Select the point which is at the maximum distance to the line of best fit of the sequence selected in S4 as a new LFDP
S6: The sequence selected in S5 is divided into two sequences by the new IDP selected in S4. And put them into the sequence list.
S7 : Return IDPs
End

Figure 3. Diagram of filter shapelet candidates based on IDPs



SAX words are masked as follows: (1) pick a character from the SAX word and (2) add this character by 1 to create a new character. The newly-generated SAX word is put into the SAX words container. This process is repeated 10 times.

Figure 4. False dismissals of SAX



4.4. Find Top *k* SAX Word

For a SAX word, a time series may contain it or not. This should be counted in advance. By doing this, the training set is divided into two datasets: (1) the time series in one dataset contains the given SAX word; and (2) the time series in the other dataset does not contain the given word. The information gain of one SAX word can be calculated as shown in equation (7).

$$gain\left(sp\right)=e\left(D\right)-\frac{n_C}{n}e\left(D_C\right)-\frac{n_N}{n}e\left(D_N\right) \tag{7}$$

In equation (7), D is the whole dataset and n_c is the number of time series in D. D_C is the dataset and contains the given SAX words, while n_c is the number of time series in D_C. D_N is the dataset and does not contain the given SAX words, while n_N is the number of time series in D_N. The entropy e is defined in Section 2.

The gain value of each SAX word will be calculated. The top k SAX words with great gain will be selected.

4.5. Discover the Best Shapelet

After the top k SAX words are identified, the best shapelet can be discovered with the help of the top k SAX words.

SAX words correspond to subsequences. After identifying the top k SAX word, mapping relation is used to find the corresponding subsequences. Next, the best shapelet from the corresponding subsequences can be discovered.

The best shapelet means that the subsequence with the maximum information gain, which broken by maximizing the separation gap will be selected as the final shapelet.

5. EXPERIMENTS

5.1. Datasets

The authors performed experiments on datasets from the UCR Time Series Classification Archive (Chen et al., 2015). Those datasets are in an “arff” file format. This can be downloaded from the UEA TSC website (Bagnall & Lines, 2015).

The UCR Time Series Classification Archive is chosen for three reasons: (1) this archive has a large number of publicly accessible datasets (Li, Bissyandé, Klein, & Le Traon, 2016); (2) these datasets cover a wide range of domains, including environmental monitoring and medical diagnosis (Li et al., 2016); and (3) the results of comparative experiments on these datasets can be found in the relevant literature.

The datasets used for experiments is show in Table 1. The training set size ranges from 20 to 1,000. The training set length ranges from 24 to 720.

5.2. Accuracy Comparison

In TSC, the nearest neighbor algorithm is an accurate and robust method. In this article, the nearest neighbor algorithm with Euclidean distance (1NN-EU) is selected as the benchmark method.

A set of comparative experiments of classification accuracy was performed among 1NN-EU (the benchmark method), FS-KP (Zhang et al., 2016), FS (Rakthanmanon & Keogh, 2013), FS-SFA (the authors’ previous work, Ji, Hu, et al., 2016), and FS-IDPs (the method proposed in this article). Accuracy rates are shown in Table 2.

As shown in Table 2, FS, FS-SFA, and FS-IDPs are more accurate than the benchmark method (1NN-EU). FS-KP and 1NN-EU are almost at the same level.

Table 2 also shows that FS-SFA (the previous work of the authors) and FS-IDP (the method proposed in this article) slightly improved classification accuracy rates compared with FS. The accuracy rates of FS-KP are reduced. To accelerate the process of shapelet discovery, FS-KP sacrificed some accuracy.

Overall, the classification accuracy rates of FS-IDPs retained a high level.

5.3. Time Consuming Comparison

Compared with FS, FS-IDPs and FS-SFA keep the same level in accuracy rates. Next, the experiments of time consuming comparison were done.

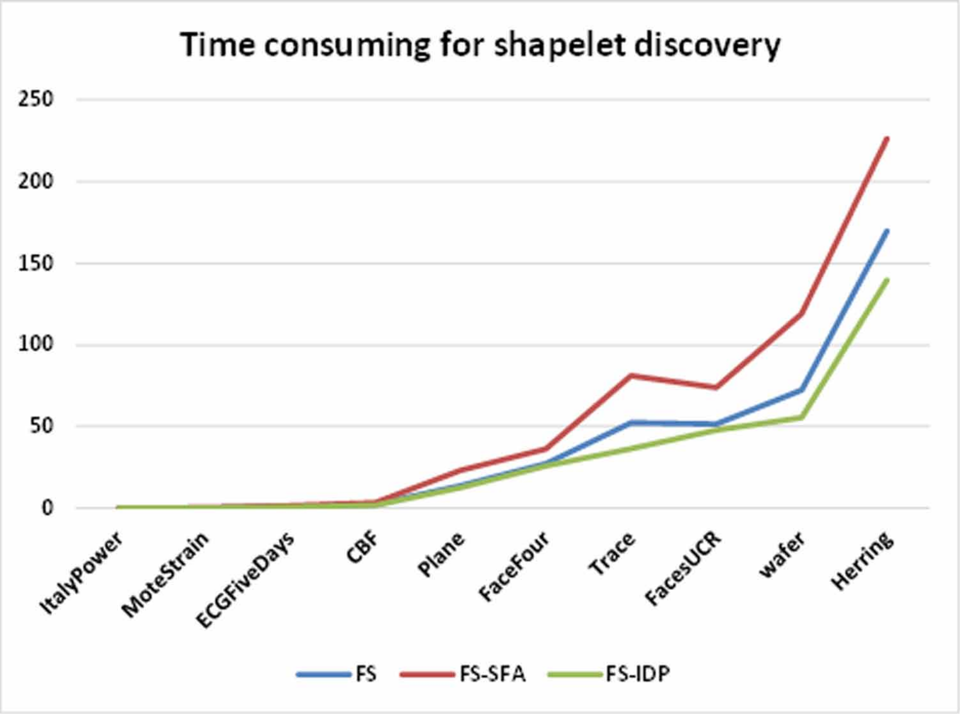
Table 1. Datasets used in experiments

Dataset	Number of Classes	Size of Training Set	Size of Testing Set	Time Series Length
ItalyPower	2	67	1,029	24
Trace	4	100	100	275
Wafer	2	1,000	6,174	152
MoteStrain	2	20	1,252	84
Plane	7	105	105	144
CBF	3	30	900	128
FaceFour	4	24	88	350
FacesUCR	14	200	2,050	131
ECGFiveDays	2	23	861	136
Herring	2	64	64	512

Table 2. Accuracy rates comparison showing different methods and different datasets

Dataset	1NN-EU	FS-KP	FS	FS-SFA	FS-IDPs
ItalyPower	0.955	0.896	0.917	0.923	0.919
Trace	0.760	0.990	1.000	0.990	0.980
Wafer	0.995	0.998	0.997	0.998	0.987
MoteStrain	0.879	0.818	0.777	0.835	0.831
Plane	0.962	1.000	1.000	0.981	0.990
CBF	0.852	0.918	0.940	0.972	0.942
FaceFour	0.784	0.818	0.909	0.932	0.920
FacesUCR	0.769	0.577	0.706	0.683	0.727
ECGFiveDays	0.797	0.772	0.998	0.999	0.998
Herring	0.516	0.500	0.531	0.578	0.563
Average	**0.827**	**0.829**	**0.878**	**0.889**	**0.886**

Figure 5. Time consuming comparisons



The plots in Figure 5 and Table 3 report the shaplet discovery execution time obtained by applying the FS, FS-SFA, and FS-IDPs algorithms on different datasets. As shown in Table 3, the average time consuming of the FS-IDPs algorithm is faster than FS algorithm by about 14%.

Table 3. Time consuming comparisons

Dataset	FS(s)	FS-SFA(s)	FS-IDPs(s)	Speed Up Rate*
ItalyPower	0.053	0.108	0.051	0.029
MoteStrain	0.278	0.475	0.243	0.124
ECGFiveDays	0.978	1.590	0.846	0.135
CBF	2.314	3.326	1.874	0.190
Plane	13.850	23.137	12.692	0.084
FaceFour	27.313	36.362	25.950	0.050
Trace	52.415	81.185	36.533	0.303
FacesUCR	51.676	74.011	47.884	0.073
Wafer	72.374	119.251	55.446	0.234
Herring	169.763	226.441	139.846	0.176
Average				**0.140**

*Speed-up rate is calculated as (FS-FS-IDPs)/FS

5.4. Summary of Experimental Results

Through these two experiments, it is found that ST-IDP algorithm is the fastest method. This is because it reduces the shapelet discovery time by approximately 14.0%. At the same, ST-IDP retains the same level of classification accuracy rates.

6. CONCLUSION

There is a big cost in TSC related to time consuming in shapelet discovery. This article presented the FS-IDPs, an algorithm identifying IDPs. Only those subsequences containing at least one IDPs was selected as a shapelet candidate. Finally, the best shapelets were selected from the candidates.

It is important to note the two assumptions in the proposed algorithm. First, time consuming decreased by reducing the number of shapelet candidates and shrinking the search range. Second, IDPs selected suitable and interpretable shapelets for accuracy rates. Experiment results illustrated that the new algorithm reduced the shapelet discovery time by approximately 14.0%. It also kept the same level of classification accuracy rates.

ACKNOWLEDGMENT

The authors would like to acknowledge the support provided by the National Natural Science Foundation of China (61402263, 91546203), the National Key Research and Development Program of China (2016YFB0201405), and the special funds of Taishan scholar construction project.

REFERENCES

Bagnall, A., Bostrom, A., Large, J., & Lines, J. (2016). *The great time series classification bake off: A review and experimental evaluation of recently proposed algorithms.* Retrieved from http://www-bcf.usc.edu/~liu32/milets16/paper/MiLeTS_2016_paper_5.pdf

Bagnall, A., & Lines, J. (2015). *The UEA TSC website*. Retrieved from http://timeseriesclassification.com

Bagnall, A., Lines, J., Hills, J., & Bostrom, A. (2015). Time-series classification with COTE: The collective of transformation-based ensembles. *IEEE Transactions on Knowledge and Data Engineering*, *27*(9), 2522–2535. doi:10.1109/TKDE.2015.2416723

Bostrom, A., Bagnall, A., & Lines, J. (2015). *The UEA TSC codebase*. Retrieved from https://bitbucket.org/aaron_bostrom/time-series-classification

Chang, K.-W., Deka, B., Hwu, W.-M. W., & Roth, D. (2012, December). Efficient pattern-based time series classification on GPU. Proceedings of the 12th IEEE International Conference on Data Mining (pp. 131–140). Piscataway, NJ: IEEE. Retrieved from http://cogcomp.cs.illinois.edu/papers/undefined.pdf

Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., & Batista, G. (2015). *The UCR time series classification archive*. Retrieved from http://www.cs.ucr.edu/~eamonn/time_series_data/

Esling, P., & Agon, C. (2012). Time-series data mining. *ACM Computing Surveys*, *45*(1), 12. doi:10.1145/2379776.2379788

Fu, T. C. (2011). A review on time series data mining. *Engineering Applications of Artificial Intelligence*, *24*(1), 164–181. doi:10.1016/j.engappai.2010.09.007

Gabr, M. M., & Fatehy, L. M. (2013). Time series classification. *Journal of Statistics Applications & Probability*, *2*(2), 123–133. doi:10.12785/jsap/020205

Grabocka, J., Schilling, N., Wistuba, M., & Schmidt-Thieme, L. (2014, August). Learning time-series shapelets. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 392-401). New York, NY: ACM.

He, Q., Dong, Z., Zhuang, F., Shang, T., & Shi, Z. (2012, December). Fast time series classification based on infrequent shapelets. *Proceedings of the 11th International Conference on Machine Learning and Applications (ICMLA '12)* (*Vol. 1*, pp. 215-219). Washington, DC: IEEE Computer Society. doi:10.1109/ICMLA.2012.44

Hills, J., Lines, J., Baranauskas, E., Mapp, J., & Bagnall, A. (2014). Classification of time series by shapelet transformation. *Data Mining and Knowledge Discovery*, *28*(4), 851–881. doi:10.1007/s10618-013-0322-1

Hou, L., Kwok, J. T., & Zurada, J. M. (2016, February). Efficient learning of timeseries shapelets. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)* (pp. 1209-1215). Retrieved from https://www.cse.ust.hk/~jamesk/papers/aaai16c.pdf

Ji, C., Hu, Y., Liu, S., Yang, C., Pan, L., & Wu, L. (2016, November). A fast shapelet discovery algorithm with symbolic fourier approximation. *Proceedings of the 6th International Conference on the Internet of Things*. New York, NY: ACM.

Ji, C., Liu, S., Yang, C., Wu, L., Pan, L., & Meng, X. (2016, May). A piecewise linear representation method based on importance data points for time series data. *Proceedings of the 2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD 2016)* (pp. 111-116). New York, NY: IEEE. doi:10.1109/CSCWD.2016.7565973

Keogh, E., & Kasetty, S. (2003). On the need for time series data mining benchmarks: A survey and empirical demonstration. *Data Mining and Knowledge Discovery*, *7*(4), 349–371. doi:10.1023/A:1024988512476

Li, D., Bissyandé, T. F., Klein, J., & Le Traon, Y. (2016, October). DSCo-NG: A practical language modeling approach for time series classification. In H. Boström, A. Knobbe, C. Soares, & P. Papapetrou (Ed.), *Advances in Intelligent Data Analysis XV, International Symposium on Intelligent Data Analysis* (pp. 1-13). Springer International Publishing. doi:10.1007/978-3-319-46349-0_1

Lin, J., Keogh, E., Wei, L., & Lonardi, S. (2007). Experiencing SAX: A novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, *15*(2), 107–144. doi:10.1007/s10618-007-0064-z

Lines, J., Davis, L. M., Hills, J., & Bagnall, A. (2012, August). A shapelet transform for time series classification. *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 289-297). New York, NY: ACM. doi:10.1145/2339530.2339579

Mueen, A., Keogh, E., & Young, N. E. (2011, August). Logical-shapelets: An expressive primitive for time series classification. *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1154-1162). New York, NY: ACM. dOI: doi:10.1145/2020408.2020587

Nunes, L., Estrella, J., Nakamura, L., de Libardi, R., Ferreira, C., Jorge, L., . . . Reiff-Marganiec, S. (2016). A distributed sensor data search platform for internet of things environments. *International Journal of Services Computing*. arXiv:1606.07932

Prieto, O. J., Alonso-González, C. J., & Rodríguez, J. J. (2015). Stacking for multivariate time series classification. *Pattern Analysis & Applications*, *18*(2), 297–312. doi:10.1007/s10044-013-0351-9

Rakthanmanon, T., & Keogh, E. (2013, May). Fast shapelets: A scalable algorithm for discovering time series shapelets. In J. Ghosh, Z. Obradovic, J. Dy, Z-H. Zhou, C. Kamath, & S. Parthasarathy (Eds.), *Proceedings of the 13th SIAM International Conference on Data Mining* (pp. 668-676). Philadelphia, PA: SIAM. doi:10.1137/1.9781611972832.74

Sampaio, A., Lima, R. C. Jr, Mendonça, N. C., & Filho, R. H. (2013). Implementation and empirical assessment of a web application cloud deployment tool. *International Journal of Cloud Computing*, *1*(1), 40–52.

Shah, M., Grabocka, J., Schilling, N., Wistuba, M., & Schmidt-Thieme, L. (2016, March). Learning DTW-shapelets for time-series classification. *Proceedings of the 3rd IKDD Conference on Data Science* (p. 3). New York, NY: ACM.

Wei, L., Keogh, E., & Xi, X. (2006, December). SAXually explicit images: Finding unusual shapes. *Proceedings of the Sixth International Conference on Data Mining (ICDM'06)* (pp. 711-720). New York, NY: IEEE. doi:10.1109/ICDM.2006.138

Wistuba, M., Grabocka, J., & Schmidt-Thieme, L. (2015). Ultra-fast shapelets for time series classification. *Journal of Data & Knowledge Engineering.* arXiv:1503.05018

Ye, L., & Keogh, E. (2009). Time series shapelets: A new primitive for data mining. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 947-956). New York, NY: ACM. doi:10.1145/1557019.1557122

Ye, L., & Keogh, E. (2011). Time series shapelets: A novel technique that allows accurate, interpretable and fast classification. *Data Mining and Knowledge Discovery*, *22*(1-2), 149–182. doi:10.1007/s10618-010-0179-5

Zhang, Z., Zhang, H., Wen, Y., & Yuan, X. (2016). Accelerating time series shapelets discovery with key points. In L. Feifei, K. Shim, K. Zheng, & G. Liu (Eds.), *Proceedings of the 18th Asia-Pacific Web Conference Web Technologies and Applications* (pp. 330-342). Switzerland: Springer International Publishing. doi:10.1007/978-3-319-45817-5_26

Cun Ji is a PhD candidate of Shandong University. His current research interests include time series data analysis, enterprise services computing and services system for manufacturing.

Chao Zhao is a master student at School of computer science and technology at Shandong University, China. He is a member of the HCI&VR Group of Shandong University. His main research interests include time series data analysis, cloud computing, etc.

Li Pan obtained her BS, MS and PhD degrees from Shandong University, China. She is a Lecturer of Shandong University. Her current research interests include cloud computing, cloud manufacturing, and market-oriented resource allocation.

*Shijun Liu obtained his BS degree in oceanography from Ocean University of China, and MS and PhD degrees in computer science from Shandong University, China. He is a Professor of Shandong University. His current research interests include services computing, enterprise services computing and services system for manufacturing.*Corresponding author.*

Chenglei Yang obtained his BS, MS and PhD degrees from Shandong University, China. He is a Professor of Shandong University. His current research interests include human computer interaction and virtual reality.

Lei Wu obtained her BS degree from Qingdao University of Science & Technology, and PhD degree from Shandong University. She is an adjunct professor of Shandong University. Her current research interests include service computing, manufacturing cloud and cloud computing.