Learning Perceptual Position-aware Shapelets for Time series Classification

Xuan-May Le[1⋆], Minh-Tuan Tran[2⋆], and Van-Nam Huynh(✉)[1]

[1] School of Knowledge Science, Japan Advanced Institute of Science and Technology, Japan
{xuanmay, huynh}@jaist.ac.jp
[2] School of Computing, Korea Advanced Institute of Science and Technology, Korea
tmtuan@kaist.ac.kr

Abstract. Shapelets are time series subsequences that effectively distinguish time series classes. Recently, time series classifiers based on shapelets have gained interest from the community thanks to their high accuracy and interpretable results. However, these shapelet-based methods still have some problems in both shapelet initialization and learning shapelet phases that limit their performances. In this paper, we propose a novel shapelet-based classifier, called Perceptual Position-aware Shapelet Network (PPSN), to effectively discover and optimize the shapelets. Our method effectively utilizes the perceptually important points to extract a small number of high-quality shapelet candidates and leverages the position-aware subsequence distance for evaluating these candidates. In the learning shapelet phase, our model applied the fixed normalization on each shapelet's transformed values to address the negative impact of their different value ranges. It also uses the stop-gradient connection in the first few epochs to reduce the unwelcome effect of the non-optimal weights of the final linear layer. Experimental results on 112 UCR datasets demonstrate that our model is state-of-the-art compared to existing non-ensemble methods and competitive with the current most accurate classifier, HIVE-COTE 2.0, while retaining the advantage of low computational time and the power of interpretation.

Keywords: Time series classification · shapelet discovery · efficiency

1 Introduction

Time series classification (TSC) has been receiving a great attention from the research community due to its importance in many real-world applications. In 2009, Ye et al. [1] introduced a novel concept of shapelets for TSC. Intuitively, shapelets are time series subsequences that can effectively discriminate the classes. It has been demonstrated to be a superb success in leveraging for TSC tasks since various classes are generally recognized by local patterns

⋆ Equal contribution

rather than global structures. Furthermore, shapelet-based approaches can convey interpretable results. Two first shapelet-based classifiers searched all possible shapelet candidates from the dataset and selected the final shapelets by their information gain. After that, they build the shapelet decision tree with the optimal shapelet at each of their nodes [1] or transform a target time series by its distance to shapelets and then leverage several standard methods to classify transform values instead of the original time series [6]. On the other hand, shapelets and learning algorithms have been integrated into recent research [2,3,4] to directly train shapelets that can identify time series of distinct classes.

In general, almost the shapelet-based classifiers contain two main phases: (i) shapelet initialization phase that discovers shapelet candidates from training time series and then selects the final shapelets by using quality evaluations like information gain, Kruskal-Wallis statistic or F-statistic; (ii) learning shapelet phase that optimizes shapelets through a gradient descent algorithm with the neural network model. Nevertheless, there are shortcomings in both phases of the existing shapelet-based methods.

In the shapelet initialization phase, the first shapelet-based methods [1,6] use a so-called Full Extractor to find all possible shapelet candidates and then utilize the Euclidean Distance to define the distance between the shapelet candidates and the target time series, it yields a good performance while suffering from high computational complexity. To avoid this problem, in [2], the authors proposed to use the Fixed-Length Extractor that only draws out all the candidates of the same length. Then, they apply k-Mean for clustering these candidates and use the k-Mean centroids as initial shapelets. However, this means that the shapelet length must be fixed while there can be shapelets of different lengths in the dataset. Recently, several attempts have been made to shorten the time for the process of shapelet extractor and automatically tune cumbersome parameters (e.g., the number and length of shapelets) by exploiting piecewise aggregate approximation (PAA), potentially resulting in loss of detailed data characteristics [7,8,5]. On the other hand, leveraging Euclidean Distance to calculate subsequence distance between shapelet candidates and original long time series instances takes significant time and inadvertently ignores the position of shapelets.

In the learning shapelet phase, the previous methods learn directly from the subsequence distances (transformed values) of shapelets and target time series. However, this makes the model challenging to train and converge when some shapelets give an extremely high values, and others provide values that are considered small. On the other hand, during the first training epochs, the linear layer in the final network usually generates very unsatisfactory predictions due to its non-optimal weights. Both problems may limit the model's performance.

In this paper, we propose a novel method called the Perceptual Position-aware Shapelets Network (PPSN) for time series classification in order to tackle the aforementioned issues. In the shapelet initialization phase, we construct a perceptual shapelet extractor that automatically extracts a few prominent shapelet candidates by using three consecutive important points. Next, we use

position-aware subsequence distance for shapelet evaluation, which calculates the distance of the shapelet candidate and its corresponding time series by leveraging its position information rather than comparing it with the entire original time series, leading to both achieving better performance and reducing the computation time. Finally, high-quality shapelets with various lengths are retained effectively. In the learning shapelet phase, our model applied the fixed normalization on each shapelet's transformed values to address the negative impact of their different value ranges. Furthermore, the proposed method uses the stop-gradient connection in the first few epochs (stop-gradient epochs) to reduce the unwelcome effect of the non-optimal weights of the final linear layer.

Our contributions can be summarized as follows: (i) We propose the novel shapelet-based approach, PPSN, that combines the effective shapelet extractor and effectively applies position information to calculate subsequence distance; (ii) We also introduce the fixed normalization and stop-gradient epochs techniques that increase the model's accuracy; (iii) Extensive experiments on 112 UCR datasets show that our PPSN achieves state-of-the-art performance to non-ensemble methods while still having the power of interpretation and low computational time.

2 Relative works

2.1 State-Of-The-Art Time Series Classifiers

During the late decades, numerous algorithms have been developed for TSC, among them the ensemble-based and feature-based methods are currently the state-of-the-art. Some popular ensemble-based methods include HIVE-COTE [13] and the most accurate TSC model, HIVE-COTE 2.0 [12], that combine four highly different classifiers, each of which is designed to capture a separate discriminatory feature. InceptionTime [17] is the best deep learning model for TSC. It is proposed to reduce the variance of the sub-model by using an ensemble of five Inception-based convolutional neural networks. On the other hand, many feature-based methods have been demonstrated to be successful in the task. Specifically, ROCKET [15] feeds the target time series into convolutional kernels and then classifies their transformed features by a simple linear classifier, e.g ridge regression. MiniRocket [14] is a reformulated version of ROCKET, which makes a few adjustments in their kernels and highly optimizes the convolutional process. Thus, MiniROCKET achieves the state-of-the-art compared to non-ensemble classifiers. Shapelet-based classifiers [1,6] are also the feature-based approaches. However, since essential local patterns (shapelets) obtained from original time series are employed to designate their class, shapelet-based approaches can deliver more interpretable decisions.

2.2 Perceptually Important Points

Perceptually Important Points (PIPs) method was first proposed [9] in order to extract important points from a price series. PIPs are then mainly used in

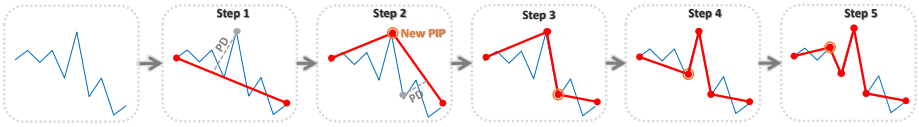


Fig. 1: The process of extracting first 6 PIPs. In that, PD is calculated by Eq. 1.

time series data mining for many tasks such as data representation or dimension reduction. Assume we have a time series $T = [t_1, ..., t_n]$, where k is the number of salient points to be extracted. To begin, we create a list of PIPs ($PIPs = [1, n]$) with the first and last indexes of T. The maximum Perpendicular Distance (PD) from the line created from two preceding elements added to PIPs is then determined by recursively finding the index in T with the highest PD. Eq. 1 is used to compute the PD between one position *pos* and PIPs.

$$PD(pos, PIPs) = \frac{a * P_{pos} - T_{pos} + c}{\sqrt{a^2 + 1}} \tag{1}$$

where, $a = \frac{T_e - T_s}{P_e - P_s}$, $c = T_e - a * P_e$ and $P = z\_norm([1, ..., n])$ is a z-normalized list of positions. In which, given g where $1 \leq g \leq k$ and $PIPs_g < pos < PIPs_{g+1}$, define $s = PIPs_g$ and $e = PIPs_{g+1}$. The Fig. 1 shows an example of finding the first six PIPs from the target time series.

3 Preliminaries

In this section, we provide all of the essential definitions and notations.

Definition 1. Time Series. *A time series T is a sequence of real numbers collected at regular intervals over a period of time: $T = [t_1, .., t_n]$, where $n \in N$ is length of T.*

Definition 2. Time Series Dataset. *A time series dataset D consists of m time series: $D = [T_1, ..., T_m]$, where T_i is the $i-th$ time series in D with Y_i is label of T_i. Note that, $Y_i \in Y$ is label of dataset and $|Y|$ is indicated as number of classes in dataset D.*

Definition 3. Subsequence. *Given a time series T of length n, a time series subsequence $T_{i,i+l-1} = [t_i, ..., t_{i+l-1}]$ is a consecutive subsequence of time series T, where i is a starting position and l is length of S with $l \leq n$.*

Definition 4. Time Series Distance Measure. *Time series distance measure is a crucial function for determining the similarity of two time series.*

Complexity Invariant Distance. *Complexity Invariant Distance (CID) [18] is motivated on the notion that complex time series are frequently seen to be more comparable to simple time series than to other complex time series. The*

complexity-invariant estimate, $CI(Q) = \sqrt{\sum_{i=1}^{n-1}(q_{i+1} - c_i)^2}$*, is used to calculate the* CID *of* Q *and* C *as follows:*

$$CID(Q, C) = ED(Q, C) * \frac{max(CI(Q), CI(C))}{min(CI(Q), CI(C)}, \tag{2}$$

where ED is Euclidean Distance:

$$ED(Q, C) = \sqrt{\sum_{i=1}^{n}(q_i - c_i)^2} \tag{3}$$

Definition 5. Subsequence Distance (SubDist). *Given a time series* $T = [t_1, ..., t_n]$ *of length* n*, and a subsequence* $S^i = [s_1^i, ..., s_l^i]$ *of length* l*, with* $l \leq n$*, the subsequence distance of* T *and* S^i *(SubDist) is determined as:*

$$SubDist(T, S^i) = \min_{j=1}^{n-l+1} \left(ED(T_{j,j+l-1}, S^i)\right) \tag{4}$$

In this work, we utilize the CID to calculate the SubDist between T and S^i, called CID_SubDist, the formulation for determining CID_SubDist is given by:

$$CID\_SubDist(T, S^i) = \min_{j=1}^{n-l+1} \left(CID(T_{j,j+l-1}, S^i)\right) \tag{5}$$

Definition 6. Information Gain (Infogain). *Given a time series dataset* D *with two labels* A *and* B*, where* $p(A)$ *and* $p(B)$ *represent the percentage of instances in each class. Given a split strategy sp that divides* D *into two sub-datasets* $D1$ *and* $D2$*. This splitting's information gain is determined as follows:*

$$IG(sp) = E(D) - \left(\frac{|D_1|}{|D|}E(D_1) + \frac{|D_2|}{|D|}E(D_2)\right) \tag{6}$$

where $|D|$ *denotes the number of instances in dataset* D*, and* $E(D)$ *denotes the entropy of* D*, which is calculated as follows:*

$$E(D) = -p(A)log(p(A)) - p(B)log(p(B)) \tag{7}$$

Definition 7. Optimal Split Point (OSP). *Give time series dataset* D *and a subsequence* S^i*, we first compute SubDist between* S^i *and all instances of* D*, and then sort the distance collection. For separating* D *into* D_1 *and* D_2*, we pick certain distance thresholds* d_t*. For instance,* $SubDist(S^i, T_i) \leq d_t$ *if* $T_i \in D_1$*, while* $SubDist(S^i, T_i) > d_t$ *if* $T_i \in D_2$*. An Optimal Split Point (OSP),* $OSP(S^i)$*, is a set of thresholds with the best information gain when compared to other thresholds* d_t^**.*

$$IG(S^i, OSP(S^i)) \geq IG(S^i, d_t^*) \tag{8}$$

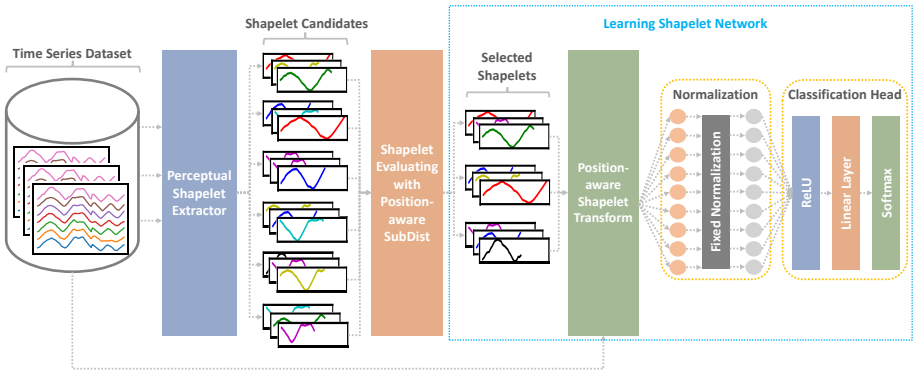


Fig. 2: General Architecture of Perceptual Position-aware Shapelet Network.

Definition 8. Shapelet. *Given a shapelet candidate S^i of class Y_i with its corresponding $OSP(S^i)$. It is considered as a shapelet when it has the highest information gain compared to any other candidates S^*.*

$$IG(S^i, OSP(S^i)) \geq IG(S^*, OSP(S^*)) \tag{9}$$

As a result, it can discriminate a class Y_i from other classes $Y \backslash \{Y_i\}$.

Definition 9. Soft-minimum Function. *Arcoding to [2], the minimum functions in 4 and 5 are not differentiable. We therefore use the soft-minimum function instead of original minimum function. Given a time series T of length n, and a shapelet S^i of length l. The CID_SubDist of T and S^i is calculated as Eq. 10. In that, when $\alpha \to \infty$ the soft-minimum approaches the true minimum.*

$$CID\_SubDist(T, S^i) = \frac{\sum_{i=1}^{n-l+1} CID_{i,l} e^{\alpha CID_{i,l}}}{\sum_{i=1}^{n-l+1} e^{\alpha CID_{i,l}}} \tag{10}$$

where, $CID_{i,l} = CID(T_{i,i+l-1}, S^i)$

4 Perceptual Position-aware Shapelet Network

In this section, we propose the novel Perceptual Position-aware Shapelet Network (PPSN). Specifically, our method uses the Perceptually Important Points to extract the shapelet candidates (Perceptual Shapelet Extractor at Section 4.1) and leverages the position information on calculating SubDist (Position-aware SubDist at Section 4.2) for evaluating these shapelet candidates. Then, we also introduce two techniques for better classifying, namely Fixed Normalization (Section 4.3) and Stop-Gradient Epochs (Section 4.3). The general architecture of PPSN is shown in Fig. 2.

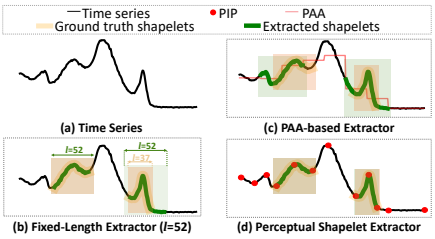


Fig. 3: Shapelets in Beef dataset selected by compared extractors. Ground truths are the most infogain shapelets.

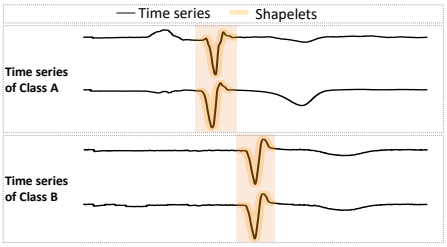


Fig. 4: Two classes in CinCECGTorso dataset have a same shapelet in different positions.

Table 1: Average Information Gains of PSE with different number of continuous PIPs on 10 first UCR datasets.

Number of continuous PIPs	2	**3 (Default)**	4	5	Full Extractor
Avg. Information Gain	0.501	**0.631**	0.601	0.581	0.652

4.1 Perceptual Shapelet Extractor

Extracting shapelet candidates is the most critical component of shapelet-based classifiers. From that, the high-quality shapelets can increase the model's performance [7,5]. However, the current extractors have their own problems. The Full Extractor used in [1,6], for instance, draws out all possible candidates from the dataset that can provide the highest quality after evaluation, but its complexity is the significant problem. To avoid the problem, Fixed-Length Extractor [2] draws out the same length l candidates. However, the method requires the length of shapelet as its parameter, while finding the optimal fixed length is challenging. Furthermore, time series often has shapelets with various lengths; therefore, constraining shapelets' length can hurt the accuracy. For instance, the most infogain Fixed-Length Extractor (with $l = 52$) cannot draw out the shapelets that perfectly cover the second ground truth of length 37. Note that, the ground truths are the most infogain shapelets extracted by Full Extractor. [8,7,5] proposed to use the PAA-based extractor to reduce the complexity. However, this may make the methods suffer from the loss of detailed data characteristics. In Fig. 3(c) the extracted shapelet is bigger on both sides compared to the ground truths since they only use the reduced-information segments.

We propose to use Perceptual Shapelet Extractor (PSE), which leverages the PIPs to efficiently pick out the high-quality shapelet candidates of various lengths. We conduct the experiment in Table 1 to show that 3 continuous PIPs provide the highest infogain which is close to Full Extractor's score (0.631 compared to 0.652). The Algorithm 1 shows the pseudo-code of PSE. Specifically, with each new extracted PIPs, p, three new possible candidates are checked and added into the candidate pool if they exist (Line 9 $\rightarrow$ 15). Fig. 3(d) demonstrates that our PSE can extract perfectly similar shapelets with ground truths.

Algorithm 1 Perceptual Shapelet Extractor

Input: Time series data set: $D = [T_1, ..., T_m]$, number of important points: k, and n is length of all time series in D
Output: Shapelet candidates set SCs, candidates' start position set SC_start_pos, and candidates' end position set SC_end_pos

$SCs = SCs\_start\_pos = SCs\_end\_pos = []$
for $i = 1$ to $|D|$ **do**
    PIPs = [1, n]
    for j = 1 to k - 2 **do**
        Find p from 1 to n with max $PD(T_i[pos], PIPs)$ where $p \notin PIPs$
        $PIPs$.append(p)
        $PIPs$.sort()
        for $z = 0$ to 2 **do**
            # Check if the candidate is valid, if yes add it into SCs
            if $p - z \geq 1$ **and** $p + 2 - z \leq |PIPs|$ **then**
                $start\_pos = PIPs[p - z]$, $end\_pos = PIPs[p + 2 - z]$
                SCs.append($T_i[start\_pos : end\_pos]$)
                SCs_start_pos.append($start\_pos$)
                SCs_end_pos.append(end_pos)
return SCs, SCs_start_pos, SCs_end_pos

4.2 Position-aware Sub-Distance for Shapelet Evaluating

Position-aware Sub-Distance (PSD). Subsequence distance (SubDist) is the distance of shapelet and the best matching location in the time series instance. Typically, ED is used to calculate the SubDist of the shapelet candidate to the entire target time series, which ignores the position information of the shapelet. As a result, it significantly elevates their computing cost and renders them susceptible when the major difference between time series of various classes is the location of the shapelet. In Fig. 4, two first time series belong to the same class A, and two last ones come to class B. Obviously, four time series share a similar subsequence, but differ in their occurrence position. To address this issue, we use a Position-aware SubDist (PSD), which only computes the SubDist between the shapelet and the subsequence in target time series with the original position of the shapelet, but enlarges on both sides with a window size w. Given time series T of length n, shapelet S^i with its start position s_i and end position e_i, and window size w. The Position-aware SubDist (PSD) of T and S^i is calculated as Eq. 11. Note that, instead of ED, we use the CID for the SubDist (as Eq. 5).

$$PSD(T, S^i) = CID\_SubDist(T[s\_pos_i : e\_pos_i], S^i), \tag{11}$$

$$s\_pos_i = \begin{cases} s_i - w + 1, & s_i - w + 1 \geq 1 \\ 1, & \text{otherwise} \end{cases}; \quad e\_pos_i = \begin{cases} e_i + w, & e_i + w \leq n \\ n, & \text{otherwise} \end{cases}$$

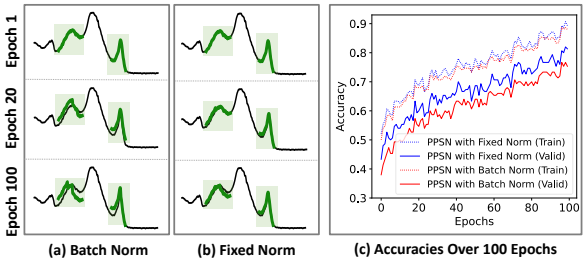


Fig. 5: (a,b) The changing of shapelets after 1, 20 and 100 epochs. (c) Average accuracies over 5 runs in Beef dataset of compared methods.

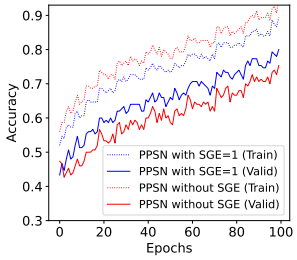


Fig. 6: Average accuracies over 5 runs in Beef dataset of compared methods.

Shapelet Evaluating with PSD. Given the shapelet candidates set $SCs = [S^1, ..., S^c]$, we find OSP of them with PSD between each S^i and all instances of D. Then, the top highest infogain g shapelet candidates are considered as selected shapelets $S = [S^i, ..., S^g]$. Given S^* is any instance in $SCs \backslash S$.

$$IG(S^i, OSP(S^i)) \geq IG(S^*, OSP(S^*))$$

4.3 Learning Shapelet Network

The ground truth shapelets have better discriminant capabilities that may not occur in the training time series instance. In this module, we use the set of selected shapelets as the learnable parameters and try to optimize it by the learning shapelet network. The model includes four modules: Position-aware Shapelet Transform, Fixed Normalization, Classification Head, and Stop-Gradient Epochs.

Position-aware Shapelet Transform. With the utility of PSD mentioned at Section 4.2, our method transforms the time series by the PSD (Eq. 11) instead of original SubDist. Given the set of shapelet $S = [S^i, ..., S^g]$ and the input time series T. The transformed vector, $Z = [Z_i, ..., Z_g]$ of T is computed as follows:

$$Z_i = PSD(T, S^i), \quad \forall i \in [1, .., g] \tag{12}$$

Fixed Normalization. Each shapelet has a different OSP to classify its class and others. This makes the shapelet generate the different ranges of SubDist values. Consequently, it makes the model challenging to train and converge when some shapelets generate extremely high SubDist values, and others provide significantly small values. We proposed the fixed normalization on each shapelet's transformed values to address this problem. Given the vector of transformed Position-ware SubDist of Shapelet S^i over a mini-batch: $\mathcal{B} = [Z_i^1, ..., Z_i^b]$ with b is number of time series instances in the batch. The normalization vector

$\bar{\mathcal{B}} = [\bar{Z}_i^1, ..., \bar{Z}_i^b]$ of Z_i is calculated as follows:

$$\bar{Z}_i^j = 1 - \frac{Z_i^j}{\sigma}, \quad \forall j \in [1, .., b] \tag{13}$$

where $\sigma = max(\sigma, max(Z_i))$ is the learned parameter. Unlike batch normalization, σ is only updated on the first epoch. In Fig. 5, by stopping the σ update, the shapelets of PPSN with fixed normalization are not changing excessively, while its accuracies are higher than that of batch normalization. It demonstrates the utility of fixed normalization.

Classification Head. After normalizing the transform value, we use the simple neural network containing a ReLU activation and a single Linear Layer to optimize the shapelets. Softmax function is then used for calculating the predicted label. Given the normalization vector $V = [V_1, ..., V_g]$, the predicted label $\hat{y} = [\hat{y}_1, ..., \hat{y}_{|Y|}]$ is predicted as follows:

$$h_i = W_{i,0} + \sum_{j=1}^{g} W_{i,j} ReLU(V_j), \quad \forall i \in [1, .., |Y|] \tag{14}$$

$$\hat{y}_i = \frac{e^{h_i}}{\sum_{j=1}^{|Y|} e^{h_j}}, \quad \forall i \in [1, .., |Y|] \tag{15}$$

where $W_{i,j}$ and $W_{i,0}$ denote the weights and bias of Linear Layer respectively, and ReLU is the activation function computed as follows: $ReLU(V_j) = max(0, V_j)$. We use the cross-entropy loss function for this model:

$$Loss = -\sum_{i=1}^{|Y|} y_i * log(\hat{y_i}) \tag{16}$$

Stop-Gradient Epochs. Our Perceptual Shapelet Extractor provides very high-quality shapelets for classifying time series. However, during the first training epochs, the linear layer in the final network usually generates very unsatisfactory predictions due to its non-optimal weights. Therefore, we apply the stop-gradient epochs for our model in which the shapelet is not updated. As can be seen in Fig. 6, PPSN without SGE has a significant drop in validation accuracies in the first few epochs, it makes the gap of the model far bigger than that of PPSN with SGE. That demonstrates the negative impact on the accuracy of non-optimal weights and the benefit of stop-gradient epochs.

5 Experimental Results

In this work, we follow to [12] perform experiments on 112 datasets UCR Time Series Archive [10] in the original train/test split (which does not include unequal

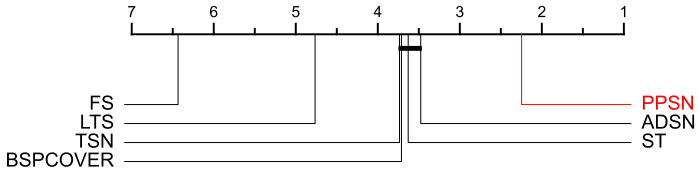


Fig. 7: Critical different diagram shows the average ranks of PPSN and 6 shapelet-based methods on 85 UCR Dataset. Solid lines indicate the group in which there is no significant difference (p-value > 0.05)

length and missing values datasets). They vary by the dataset types, number of classes, number of instances, and time series lengths.

To compare multiple classifiers on multiple datasets, we follow the recommendation in [11] and report the result on a critical different diagram that contains average ranks instead of error rates. A black horizontal line connects methods whose pairwise classification accuracy difference is not statistically significant using a two-sided Wilcoxon signed-rank test ($\alpha = 5\%$). Holm correction is used as the post-hoc test to the Friedman test [11] for all comparison.

In order to reproduce our experiments, we built a website [3] containing all results on 112 UCR datasets and the source code.

5.1 Hyperparameter Setting

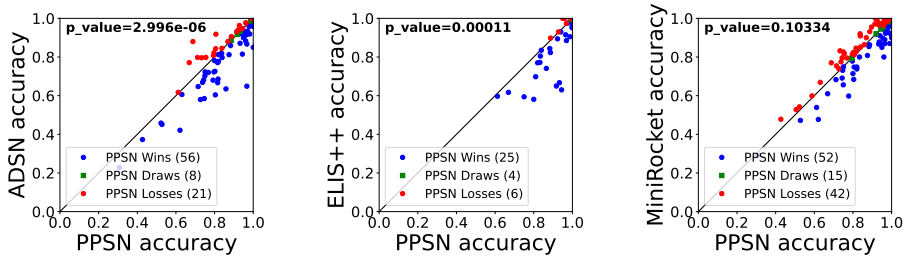
The number of Stop-Gradient Epochs and PIPs are fixed at 1 and 0.3 of time series length, respectively. The number of shapelets g is searched over $\{0.1, 0.2, 0.5, 1, 2, 5, 10\}$ of time series length. We use the simple heuristic approach for searching the window size w from $\{5, 10, 20, 30, 50, 100, 200\}$. In that, with each number of shapelets g, we calculate information gain for shapelet candidates for all $w \in \{5, 10, 20, 30, 50, 100, 200\}$. We then choose the window size w, which has the highest average information gain of top g selected shapelets. From that, PPSN has only one parameter g that needs to tune.

We conduct the experiments on Pytorch and use the AdamW optimizer with learning rate at 0.01 and momentum at 0.9. For all datasets, we use the Smoothing Label at 0.1, and the Batch size depends on the size of datasets. Specifically, the batch size is chosen from $\{16, 32, 64, 128, 256\}$ if the number of training instances is higher than $\{0, 100, 200, 400, 800\}$, respectively. For example, if the number of training instances is 500, the batch size is then set at 128.

5.2 Compared with Shapelet Methods

In this section, we conduct the experiment to compare our PPSN with 6 state-of-the-art shapelet-based classifiers, including Learning Time Series Shapelet (LTS) [2], Shapelet Transform (ST) [6], Fast Shapelet (FS) [8], BSPCOVER [7],

[3] `https://github.com/xuanmay2701/ppsn`



(a) PPSN vs ADSN (b) PPSN vs ELIS++ (c) PPSN vs MiniRocket

Fig. 8: Scatter charts compare the accuracy of our PPSN and ADSN, ELIS++, and MiniRocket. Each point represents the accuracy over dataset. We only conduct the comparison on the datasets they reported (85, 35, and 109 datasets for ADSN, ELIS++, and MiniRocket, respectively)

Triple-Shapelet Network (TSN) [3] and Adversarial Dynamic Shapelet Network (ADSN) [4]. We follow the protocol in [4,3,7], we only report the result on 85 UCR datasets. We do not compare to ELIS++ [5] since they only provide the results on 35/85 datasets. Fig. 7 shows the critical different diagram for comparing our model and baseline classifiers. It is clear that our PPSN achieves the highest rank and significantly outperforms all other shapelet-based classifiers. We also provide a pair-wise comparison with ADSN in Fig. 8(a) and the ELIS++ [5] in Fig. 8(b). The charts show that PPSN is superior (including equal) to ADSN and ELIS++ in most datasets (64/85 and 29/35 datasets, respectively).

5.3 Compared with Current State-Of-The-Art Methods

PPSN is compared with 7 SOTA methods including: (i) 4 ensemble-based methods HIVE-COTE (HC1) [13], HIVE-COTE 2.0 (HC2) [12], TS-CHIEF [16], InceptionTime [17]; (ii) 2 feature-based methods Rocket [15], MiniRocket [14]; (iii) interval-based algorithms DrCIF [12]. They are chosen since they are currently the most accurate approaches for time series classification.

We conduct the experiment on all 112 datasets but only report the results of 109 datasets to follow the protocol at [14]. The average rank of our model (PPSN) and other state-of-the-art classifiers is shown in Fig. 9. PPSN is more accurate than MiniRocket, InceptionTime on average, and comparatively less accurate than the most accurate existing ensemble classifiers, especially TSCHIEF, HIVE-COTE, and HIVE-COTE 2.0, although the differences are not statistically significant. However, please note that InceptionTime, HIVE-COTE, TS-CHIEF and HIVE-COTE 2.0 are ensemble methods that combime many different models including several shapelet-based classifiers. Furthermore, our model is considerably faster than those ensemble methods in terms of computational time. We also provide the scatter chart to pair-wise compare our PPSN and the best non-ensemble methods MiniRocket (see Fig. 8(c)). The chart demonstrates that

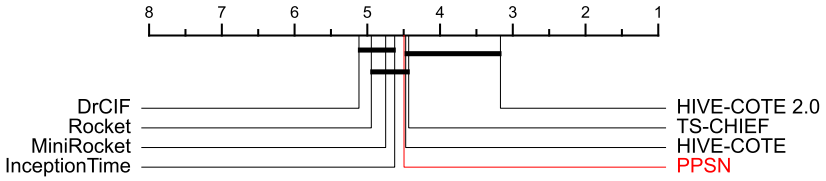


Fig. 9: Critical different diagram shows the average ranks of PPSN and 7 SOTA methods on 109 UCR datasets. Note that InceptionTime, HIVE-COTE, TS-CHIEF and HIVE-COTE 2.0 are ensemble methods that combime many different models including several shapelet-based classifiers.

Table 2: Run time (in hours) to train 109 UCR datasets. We run the shapelet initialization phase on the single thread on a cluster using AMD EPYC 7H12 2.6GHz CPU and learning shapelet phase threads on NVIDIA A40 GPU.

Methods	MiniRocket	Rocket	DrCIF	InceptionTime	HC1	HC2	TS-CHIEF
Total train time	0.25	2.85	45.4	86.58	340.21	427.18	1016.87

Our Methods	PPSN (1 thread)		PPSN (32 threads)		PPSN (64 threads)	
	Shapelet Initialization	Learning Shapelet	Shapelet Initialization	Learning Shapelet	Shapelet Initialization	Learning Shapelet
Total train time	13.73	0.62	4.23	0.62	2.47	0.62

our PPSN is superior (including equal) to MiniRocket on most datasets (68/109 datasets) with p-value at 0.10334.

5.4 Computation Time Comparison

As shown in Table 2, PPSN takes 14.35 hours to train all 109 UCR datasets, it is far faster than all of the state-of-the-art methods except Rocket and MiniRocket. Especially, PPSN is two order of magnitude faster than TS-CHIEF and 34 times faster than HC2. In addition, executing PPSN with multiple threads significantly speeds up the computational time. For instance, if we train PPSN on 32 threads and 64 threads, the running time is reduced to only 4.23 and 2.47 hours, respectively. Note that, almost of PPSN's running is taken by the shapelet initialization phase, 13.73 hours compared to only 0.62 hours of the learning shapelet phase (on single threads). This also means that the testing time of our PPSN is really fast, approximately 10 minutes for all 109 UCR datasets. The results indicate the advantage of our proposed method in terms of computational time.

5.5 Ablation Study and Sensitivity Study

We conduct several experiments on the first 30 UCR datasets to evaluate the effect of proposed components and the key parameter choice for PPSN.

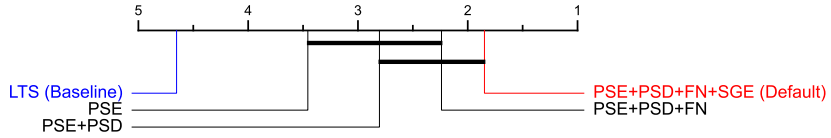


Fig. 10: Average ranks for 4 ablation versions of PPSN and LTS baseline.

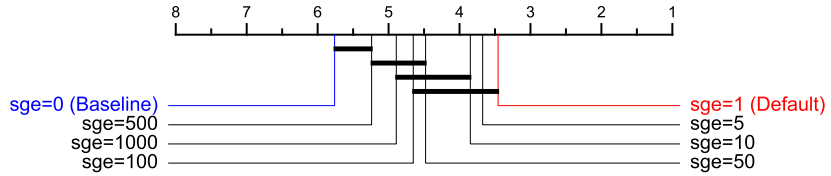


Fig. 11: Average ranks for PPSN with different number of Stop-Gradient Epochs.

Component Evaluation. We first evaluate the impact of four proposed components of our PPSN: Perceptual Shaplet Extractor at Section 4.1 (PSE), Position-aware SubDist at Section 4.2 (PSD), Fixed Normalization at Section 4.3 (FN), and Stop-Gradient Epochs at Section 4.3 (SGE). In that, the components are added one-by-one to measure their effect on final accuracy. As can be seen in Fig. 10, all four components have a positive impact on increasing the results of the final proposed model.

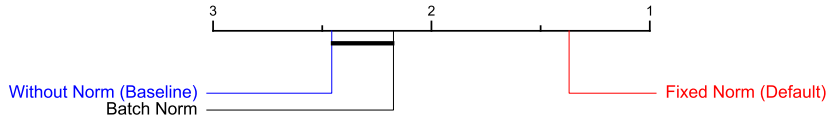


Fig. 12: Average ranks for our Fixed Normalization and Batch Normalization.

Number of Perceptually Important Points. We conduct experiments to execute our PPSN with different number of PIPs values and measure the average information gain (Eq. 4.2) of selected shapelets. We use the parameter related to the length of time series. This means that given n is length of time series, so $k = n{*}npips$. As shown in Table 3, the average of information gain of PPSN with $npips = 0.3$ (our default parameter) is approximately equal to $npips = 0.4$ and $npips = 0.5$, while the number of extracted candidates is considerably smaller. In addition, the information gain from our PPSN with $npips = 0.3$ is very close to that of Full Extractor, at 0.633 and 0.652, respectively, while only extracting 280 candidates compared to 102380 candidates of Full Extractor.

Table 3: The comparison of our PPSN with different number of PIPs.

Number of PIPs (npips)	0.1	0.2	**0.3 (Default)**	0.4	0.5	Full Extractor
Avg. Information Gain	0.592	0.611	**0.631**	0.633	0.635	0.652
Avg. No. Extracted Candidates	100.1	195.2	**280.2**	370.5	475.5	102380.9

Number of Stop-Gradient Epochs. Fig. 11 shows the effect of our PPSN model with different numbers of Stop-Gradient Epochs (SGE), ranging from 1 to 1000. While all PPSN with different number of SGE outperforms the baseline, there is almost no benefit for increasing the number of SGE. Finally, PPSN with SGE at 1 gains the most performance.

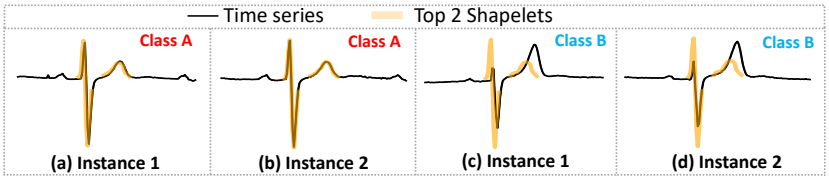


Fig. 13: Selected shapelets by PPSN for ECGFiveDays dataset.

Normalization. Fig. 12 indicates the comparison of our proposed Fixed Normalization, Batch Normalization, and Baseline (without Norm). It is clear that while there are improvements when applied normalization for PPSN, Fixed Normalization shows a significantly superior result to its counterparts.

5.6 Experiments on Interpretability

One of the great capabilities of shapelets is the power of interpretability, which can effectively provide data comprehension. Fig. 13 illustrates that the shapelets can discriminate between two classes of the ECGFiveDays dataset [10]. Electrocardiography (ECG) is a term that refers to the study of the heart. Two time series instances in Fig. 13 (a) (b) come from Class A, and those of Fig. 13 (c) (d) come from Class B. It is clear that selected shapelets by PPSN (orange lines) indicate the major differences between segments in the two classes. Specifically, the first shapelet presents a QRS complex, while the second one is a T wave of ECG. Intuitively, the T wave gained a larger peak compared to the QRS complex in Class A. In medicine, it is known as a hyperacute T wave when it occurs as a result of certain diseases such as ischemia or hyperkalemia.

6 Conclusion

This paper has proposed a novel Perceptual Position-aware Shapelet Network for time series classification, namely PPSN, including two phases. For shapelet initialization phase, we introduce an effective shapelet candidates extractor using perceptually important points and evaluate them based on position-aware subsequence distance. For learning shapelet phase, we introduce two techniques called fixed normalization and stop-gradient epochs in order to mitigate the detrimental impact of various subsequence distance ranges and diminish the unpleasant effect of the final linear layer's non-optimal weights, respectively. Our experiments show that PPSN is a state-of-the-art method compared to non-ensemble approaches. In addition, its accuracy is comparable to the current most accurate classifier, HIVE-COTE 2.0, while maintaining the benefits of low computational time and interpretive power. In future work, we intend to investigate PPSN to other time series problems such as multivariate time series classification.

References

1. Chung, F.L.K., Fu, T.C., Luk, W.P.R., Ng, V.T.Y.: Flexible time series pattern matching based on perceptually important points. In: Workshop on Learning from Temporal and Spatial Data in IJCAI (2001)
2. Grabocka, J., Schilling, N., Wistuba, M., Schmidt-Thieme, L.: Learning time-series shapelets. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 392–401 (2014)
3. Ma, Q., Zhuang, W., Cottrell, G.: Triple-shapelet networks for time series classification. In: 2019 IEEE International Conference on Data Mining (ICDM). pp. 1246–1251. IEEE (2019)
4. Ma, Q., Zhuang, W., Li, S., Huang, D., Cottrell, G.: Adversarial dynamic shapelet networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 5069–5076 (2020)
5. Zhang, H., Wang, P., Fang, Z., Wang, Z., Wang, W.: Elis++: a shapelet learning approach for accurate and efficient time series classification. World Wide Web 24(2), 511–539 (2021)
6. Lines, J., Davis, L.M., Hills, J., Bagnall, A.: A shapelet transform for time series classification. In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 289–297 (2012)
7. Li, G., Choi, B.K.K., Xu, J., Bhowmick, S.S., Chun, K.P., Wong, G.L.: Efficient shapelet discovery for time series classification. IEEE Transactions on Knowledge and Data Engineering (2020)
8. Rakthanmanon, T., Keogh, E.: Fast shapelets: A scalable algorithm for discovering time series shapelets. In: proceedings of the 2013 SIAM International Conference on Data Mining. pp. 668–676. SIAM (2013)
9. Chung, F.L.K., Fu, T.C., Luk, W.P.R., Ng, V.T.Y.: Flexible time series pattern matching based on perceptually important points. In: Workshop on Learning from Temporal and Spatial Data in International Joint Conference on Artificial Intelligence (2001)
10. Dau, H., Keogh, E., Kamgar, K., Yeh, C., Zhu, Y., Gharghabi, S., Ratanamahatana, C., Yanping, Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista &

Hexagon-ML The UCR Time Series Classification Archive. (October 2018), https://www.cs.ucr.edu/~eamonn/time_series_data_2018/
11. Dempster, A., Petitjean, F., Webb, G.I.: Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. Data Mining and Knowledge Discovery 34(5), 1454–1495 (2020)
12. Middlehurst, M., Large, J., Flynn, M., Lines, J., Bostrom, A., Bagnall, A.: Hivecote 2.0: a new meta ensemble for time series classification. Machine Learning 110(11), 3211–3243 (2021)
13. Lines, J., Taylor, S., Bagnall, A.: Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles. ACM Transactions on Knowledge Discovery from Data 12(5) (2018)
14. Dempster, A., Schmidt, D.F., Webb, G.I.: Minirocket: A very fast (almost) deterministic transform for time series classification. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. pp. 248–257 (2021)
15. Dempster, A., Petitjean, F., Webb, G.I.: Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. Data Mining and Knowledge Discovery 34(5), 1454–1495 (2020)
16. Shifaz, A., Pelletier, C., Petitjean, F., Webb, G.I.: Ts-chief: a scalable and accurate forest algorithm for time series classification. Data Mining and Knowledge Discovery 34(3), 742–775 (2020)
17. Ismail Fawaz, H., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D.F., Weber, J., Webb, G.I., Idoumghar, L., Muller, P.A., Petitjean, F.: Inceptiontime: Finding alexnet for time series classification. Data Mining and Knowledge Discovery 34(6), 1936–1962
18. Batista, G.E., Wang, X., Keogh, E.J.: A complexity-invariant distance measure for time series. In: Proceedings of the 2011 SIAM international conference on data mining. pp. 699–710. SIAM (2011)