






Transfer Learning of Shapelets for Time Series Classification Using Convolutional Neural Network

Alexandre Felipe Muller de Souza^(✉) , Mariane R. S. Cassenote ,
and Fabiano Silva 

Informatics Department, Federal University of Paraná, Curitiba, Brazil
{afmsouza,mrscassenote,fabiano}@inf.ufpr.br

Abstract. Time series classification has a wide variety of possible applications, like outlines of figure types, signs of movement and sensor signals. This diversity may present different results with the application of machine learning techniques. Among the different ways to classify time series, two, in particular, are explored in this paper: the shapelet primitive and the neural network classification. A shapelet is a sub-sequence of the time series symbolic, like a high-level descriptor, that are representative for the class to which they belong. These features act as common knowledge used by domain experts. This paper proposes a CNN training protocol to achieve better results in the classification of time series. The idea consists of decomposing the original time series into shapelets and noise. Then, the shapelets are used to train a classifier while the no-shapelets (“signal noise”) is used to train another classifier. The original time series is then used to train two final classifiers starting with the weights of shapelet and no-shapelet classifiers. This previous extraction of this representation can improve classification ability in a convolutional neural network using transfer learning. The experimental evaluation shows that the pre-selection of shapelets before the network training changes the classifier results for several databases and, consequently, improves classification accuracy.

Keywords: Machine learning · Shapelets · Time series · Transfer learning

1 Introduction

With the technology advancement and the mobile devices popularization, systems that generate real-time data have become disseminated. The amount of data generated is continuously growing and meaning attributions are needed to extract relevant information from this data. In addition to sensors, many other types of data can be represented in structures called time series. These structures can be handled as shapes, that can be projected in a one-dimensional plane or spectrogram [3]. Simply, any string of numerical values whose order forms a

sequence [10] can be called a time series. Naturally, the processing of these time series has applications in the most diverse areas of knowledge [7].

The scope of this work is limited to the problem of automated classification of time series: inputs are divided into two or more classes to be determined. In particular, the focus in this analysis is supervised classification, i.e., given a series whose classification is not known, the class to which it belongs must be inferred. This is accomplished through other examples previously classified among these classes, not exactly the same, but similar. Naturally, the series repository [3] already makes available these pre-classified (and other unclassified) examples by the domain expert.

Although there are several methods proposed for classifying time series, we will cover two of them in more details: symbolic representations, in particular the shapelet primitive extraction, and Convolutional Neural Networks (CNN). In the shapelet method, are extracted the sub-sequences of the time series that are partial and representative attribute (thus, symbolic) for the class to which they belong. The idea is to extract features from domain problem that allow to perform the classification.

In contrast to classification using symbolic representations paradigm, there are classifiers based on neural networks. These classifiers use a data model representing the memory in the form of weights of a network. Neural classifiers are known for their efficiency, as their model has already been trained, but learning is based purely on adjusting the weights of the network. This adjustment is given by an error minimization algorithm that can achieve different results depending on many factors such as network architecture, number of layers, filters, among others. CNN-based classifiers have many applications and are referred to by the term “Deep Learning” [11].

This paper proposes a pre-training method for time series classifiers using CNN. Figure 1 outlines the basic process of our method. Usually the network is started with random weights. However, in a different way, our proposal is to carry out an initial phase that extracts the relevant shapelets and separates the symbolic representations from the rest of the series, generating two new databases: one with only the parts of the series that represents shapelets, and other with only the remaining parts of the series. These databases are then used to train different models. Then a final model is generated with the original database, but starting the network with the information learned before by the previous trained models. The motivation of this approach refers to the duality between these symbolic and neural classifiers and how their performance improve by exchanging information.

Through this study, we observe that the pre-initialization of the network increases the accuracy of the final classification for most of the tested bases. There are two methods of improvement. The first consists in extracting the symbolic representations, in this case shapelets, and training only with these parts of the original series and then transferring the learned weights to the final model, which allows highlighting the symbolic representations. This case is being called “Classifier 4” in Fig. 1. Another method removes the shapelets and trains only with the “noisy part” of the original series and then apply the transfer the weights to the final model, called “Classifier 5” in the figure.

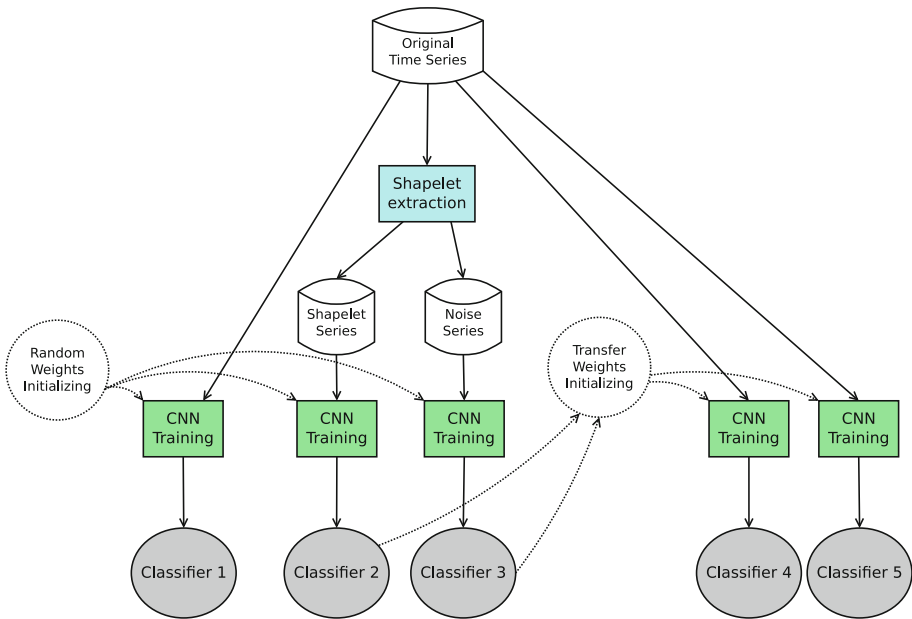


Fig. 1. Diagram of experiments.

The rest of this paper is organized as follows: related works are commented on in Sect. 2. The materials and method used are presented in Sect. 3. Section 4 contains the results and discussions, and finally, Sect. 5 concludes the paper and list some future works.

2 Related Work

In order to obtain the necessary conditions to compare classification models, researchers often use an openly available database of UCR Time Series [3]. This is a popular time series benchmark due to its diversity.

The time series classification is divided into several paradigms for generating models, whether symbolic or statistical. Most works based on symbolic paradigms use shapelets due to the intelligibility of the model. According to [20], shapelets can provide results that should help domain experts better understand their own data.

In [12] is presented an interface to visualize the extracted shapelets in an interactive way, which can also be used at different extraction methods. The first method was proposed by the same authors [13] and is based on intermediate symbolic representations, called *Symbolic Aggregate Approximation* (SAX). This concept is widely explored and was presented for the first time in [14]. It consists of discretizing the series into bands that become symbols of an alphabet and, consequently, each sub-sequence generate a word. In [13] the SAX words

are converted into shapelets through a structure of weighted bitmaps of the SAX words to determine the quality of the shapelet candidates, transforming the shapelet selection problem into a heuristic search process. According to the authors, this process makes finding shapelets more efficient.

The main shapelet-based classification methods cited in [12] are: HIVE-COTE (*Hierarchical Vote Collective of Transformation based Ensembles*) [16], ELIS (*Efficient Learning Interpretable Shapelets*) [4] and LTS (*Learning Time-series Shapelet*) [8]. In addition, the BOSS method (*Bag of SFA Symbols*) of [17] is also referenced by [19] and [6]. Each of these will be explained in detail below.

One of the most popular time series classification algorithms is called BOSS. It generates a set of symbolic SFA representations (*Symbolic Fourier Approximation*) with different sliding window sizes and creates an ensemble based on a template similar to Bag of Words (BoW). This model is a set with occurrences counting of symbols, disregarding other elements such as the order in which they appear. Compared to shapelets, these representations are extremely short and have a wide range of sizes. Its classification is based on a histogram of occurrences rather than a decision tree. The BOSS is noise tolerant, scale independent, and position invariant.

One of the works that supports several others in the field of time series classification is the use of a collection of classifiers called COTE [1]. In this type of approach there are 35 classifiers [16]. Each classifier is trained individually and then the results are combined by assigning weights to the classifiers [15]. A later variation of this method, called HIVE-COTE, is based on a hierarchical vote between classifiers of the same nature. In this method there are a total of five modules. For example, one module would be based on shapelets and another module would be based on BoW. Superior results were obtained through this hierarchical voting process. However, despite both works showing positive results, they use classifiers from different paradigms, such as based on examples, symbolic, statistical and connectionist. This makes the final classification unintelligible.

Another work related to this paper that presents a time series classifier using shapelets is ELIS [4], which has a faster extraction method than the traditional one and shows good results. This method works in two phases: shapelet discovery and shapelet adjustment. In the first stage, the generation of sub-sequences occurs in a process called PAA (*Piecewise Aggregation Approximation*). This process is like a discretization that reduces the resolution of the series. For each class the PAAs, words are ranked and this information helps to adjust the shapelet numbers to be automatically extracted. Is created a model that tries to adjust the shapelet to the best shape. The main idea is that true shapelets shouldn't appear in exactly the best shape in the training set [4]. Finally, in the second stage, several classifiers are built, each with its specific shapelets. Then a regression process is used to better adjust the shapelet in the series and generate a final classifier. ELIS presents limitations in small databases, as well as the number of parameters that need to be adjusted. In order to outline these limitations, the authors recently proposed the ELIS++ method [21], which in addition to using data augmentation, also adjusts its parameters automatically.

Another very relevant work in the area of shapelets is LTS[8]. In that work, the authors obtain the shapelets not by candidate calculations, but through a gradient descent algorithm. In this way, the process becomes very similar to adjusting the weights of a neural network.

Although efficient, all these classifiers do not completely replace the use of a classifier based on a CNN. One work that managed to take CNN comparatives in time series was [6]. In this time series classification review, the authors reproduced the main implementations and compared deep learning methods with other methods. According to the authors, HIVE/COTE is currently considered the state-of-the-art algorithm for time series classification when evaluated in the 85 datasets of the UCR file. Through the empirical study carried out by the authors, ResNet came closest to COTE/HIVE in terms of accuracy.

On the other hand, when talking about time series, is a consensus that there is a lack of research on the use of CNN in this context[2,5]. Therefore, we have an important area to be explored, as learned symbolic representations (as they are in images) offer an important intelligibility that can also be used for time series. The implementation of the CNN presented in this paper was derived from [19], which has already obtained results close to the state-of-the-art in 2016.

Considering all this context, there seems to be an interesting research line to correlate classifiers based on CNN and shapelets. The motivation for this proposal is not new. Several authors focus on this theme, although each one has its own methodology and presentation. For example, in [2] the authors assume that learning LTS shapelets is a particular case of learning in convolutional networks. The authors explain that the measure of distance from shapelets can be transformed into a convolutional form and, thus, there is a relation between both methods. However, this specific study only aims to propose a multivalued classifier based on CNN. Thus, the study justifies its positive result by the ability to learn these representations similar to shapelets. Also in this work, the authors present a theoretical formalization, correlating the layers of a CNN with shapelets and calculating the Euclidean distance. Although the authors present this theoretical formalization, they do not deepen this relationship, showing it experimentally.

Although not directly related, some other knowledge is involved with this issue. Another relevant work to be cited is [5], which explores the issue of transfer learning. This concept is realized when the weights of a network previously trained in another series are used as initial training weights, instead of randomly assigned weights. This learning process is faster and has better generalization, which is due to the transfer of representations between the original series and those that received the knowledge. The article even mentions series based on symbols, as a particular case of good generator of models. Thus, it is evident that, when transferring knowledge from this series, the convergence of the neural network is faster, reinforcing the hypothesis that the neural network model learns symbolic representations (shapes). This experiment influences the development of our proposal, as we will see in Sect. 3.

3 Materials and Method

In this section, some implementation details of the experiments performed in this work are presented. The classifier, the source code and the method used are available for consultation¹. This work aims to serve as a basis for derivative works, whether for reproduction of procedures, analysis or inspiration for other classifiers based on CNN.

The experiment that generates the classifier began by choosing the series that make up the database. The choice of the 20 series represented in Table 1 was due to their diversity, as well as to the results published in previous works of classifiers based on symbolic representations. The data are presented in text files with different formats, but already normalized and standardized. The chosen database has training and testing examples, i.e., each dataset has pre-classified examples in predetermined quantities to be used in model generation and validation.

Table 1. Selected time series from [3].

#	Dataset	Training	Validation	Length	Classes	Type
1	BeetleFly	20	20	512	2	IMAGE
2	BirdChicken	20	20	512	2	IMAGE
3	ECGFiveDays	23	861	136	2	ECG
4	ECG200	100	100	96	2	ECG
5	CBF	30	900	128	3	SIMULATED
6	FaceFour	24	88	350	4	IMAGE
7	FacesUCR	200	2050	131	14	IMAGE
8	Gun_Point	50	150	150	2	MOTION
9	ItalyPowerDemand	67	1029	24	2	SENSOR
10	Lightning7	70	73	319	7	SENSOR
11	Lightning2	60	61	637	2	SENSOR
12	MoteStrain	60	61	637	2	SENSOR
13	OliveOil	30	30	570	4	SPECTRO
14	DiatomSizeReduction	16	306	345	4	IMAGE
15	Coffee	28	28	286	2	SPECTRO
16	Symbols	25	995	398	6	IMAGE
17	Beef	30	30	470	5	SPECTRO
18	SyntheticControl	300	300	60	6	SIMULATED
19	Trace	100	100	275	4	SENSOR
20	TwoLeadECG	23	1139	82	2	ECG

¹ https://github.com/alexandrefelipemuller/timeseries_shapelet_transferlearning, last accessed 16 Aug 2021.

3.1 Shapelet Extraction

This step is the differential of this work in relation to a conventional training using a classifier based on neural networks. The extraction of shapelets is a task that usually demands a lot of processing time and neural network training. Finding these sub-sequences is an additional step that allows to transfer them to the model to be used in the original series.

Given the set of series, the most relevant shapelets from each one were extracted. For this, we used the traditional method with a sliding window and candidate dictionary [20] with sizes from 5 to 18. This choice of size was adjusted to balance the proportion of what was shapelet and what was not in similar proportions in our database. In this method, an sorted list of all representations is generated with the computation of the gain. If we had to choose only one shapelet per class, choosing the first element from this list would be enough. In our case, the three most relevant are selected, discarding intersections. This is done by looking at the position in which the shapelet was extracted in the original series: if there is a conflict in the position (however different examples are involved), the deletion occurs. One to three most relevant shapelets from these candidates were chosen. The choice criterion is based on the gain and on having no intersection for most series.

The resulting shapelets serve to generate two extra supporting databases, totaling three databases named below:

- Original series;
- Shapelet series: where only the parts corresponding to the shapelets is maintained from the original series and everything else is replaced by the central value of the series. A central value is understood as the mean value of the extremes of the complete series;
- Noise series: where only the parts that do not match the shapelets were maintained from the original series, with the part corresponding to the shapelets replaced by the central value.

The process to generate these two databases is shown as example in Fig. 2. If we ignore the parts filled by the central value, the join between the shapelet series and the noise series corresponds to the original series.

3.2 Neural Network Training

As described in Sect. 2, the one-dimensional convolution training process is performed from the three databases. The initial training is based on the implementation of [19], which does not use the concept of shapelet. The only pre-processing performed was to shuffle the training base to improve neural network training.

The network architecture has been empirically adjusted to 3 one-dimensional layers. The first convolutional layer of the network has 32 filters and kernel length 8 (window equivalent concept), the second layer has 64 filters and kernel length 5 and the third convolutional layer has 32 filters and kernel length 3. At the end, there is a pooling layer and the output corresponds to the number of classes, thus

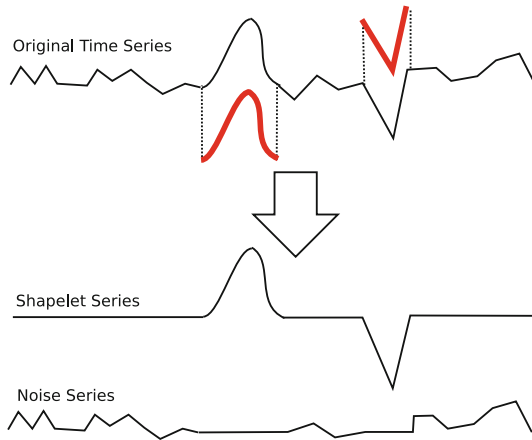


Fig. 2. Example of shapelet and noise series generation with 2 shapelets.

obtaining the probability of classification of each one. Regarding [19], the removal of “dropout” was performed. This technique consists of randomly removing some neurons to improve generalization and avoid overfitting [18].

To verify the impact of the transfer learning on classifier performance, we conducted the following experiments:

0. Process that precedes the training: extraction of shapelets and generation of support databases;
1. Original series training starting with random weights;
2. Shapelet series training starting with random weights;
3. Original series training starting with model imported from experiment 2;
4. Noise series training starting with random weights;
5. Original series training starting with model imported from experiment 4;
6. Cross validation, shapelet series training, importing model from 4;
7. Cross validation, noise series training, importing model from 2.

The objective of item 1 is to create a baseline comparison for the following tests. In this way, the results of this item should be similar to those obtained in [19]. Items 2 and 4 aim to obtain a training model formed by shapelets or by the rest of the series without shapelets (noise). Items 3 and 5 are the most important, as they aim to show the relevance of transferring the previously trained model. As all series have the same length filled with central value of the series, changing the initial model is possible. Finally, items 6 and 7 serve to elucidate and ratify the importance of transfer learning in each situation.

As the classification model is based on a neural network, if the same information is not put into training again, the weights suffer from loss of information during training. This means that if experiment 6 gives better results than 7, the shapelets are the most relevant piece of information. Thus, although all experiments have the same network architecture, only experiments 1, 3 and 5 use the complete data. All others use partial data.

3.3 Training and Transfer Learning

The extraction of symbolic representations, is performed by a deterministic algorithm, so its result is not changed in different executions. However, as the training process contains random factors related to weight adjustment, neural network training may vary. Therefore, experiments resulting from neural network training need to be statistically evaluated.

The selected databases from Table 1 are separated between training and validation (or testing). During neural network training, weights are adjusted with examples from the training base. At the end of each epoch (which corresponds to the end of the adjustment of the weights), the accuracy of the model in the validation base is verified, but this result is only a consultation. While training takes place, the error in the training base continues to be adjusted and decreased, but at the same time the error in the validation base increases. This difference is basically a form of over-adjustment.

Given the size of the time series bases, learning in 300 epochs was determined, with the best model between these epochs being saved for the validation step. For most of the tested series, the overfitting starts to happen between 150 and 250 epochs, remaining saved for the best model. We chose to run for 300 epochs because later models are discarded. As a rule, the model validated only on the training basis achieves 100% accuracy in this way. The best model is given by a checkpoint, i.e., as training continues and the error increases, the best model is saved until a better one is found. Therefore, there is no stopping criterion in this 250 epoch range because the best model is already saved.

As we transfer the model within the domain itself, it is expected that every experiment that receives a pre-trained model will rapidly evolve in number of epochs to reach a satisfactory result. The behavior in this transfer learn situation is the faster evolution of the model's accuracy [5]. For this reason, the greatest expectation in defining the experiments was that experiment 3, in addition to evolving quickly, would have better accuracy. This expectation is also due to the theoretical context exposed in Sect. 2.

4 Experimental Evaluation

As explained in Sect. 3, a total of 30 training rounds were carried out. For comparison with other works, it is expressed as means in Table 2.

The average accuracy of experiment 1 is equivalent to that presented in [19] in the databases used in both works. This validate our similar implementation of the proposed model.

Experiment 4 outperformed experiment 2 (Fig. 3). This suggests that there is still a lot of information relevant to classification that is comprised outside of extracted shapelets. In part, this is due to the size of the extracted shapelets that have absolute lengths of at most 18 values, which for most bases is a small length. An increase in this length would adjust this result, but at the cost of not making sense to highlight a very large segment. As the experiment extracts the segment from the original series, making the shapelet segment too large would

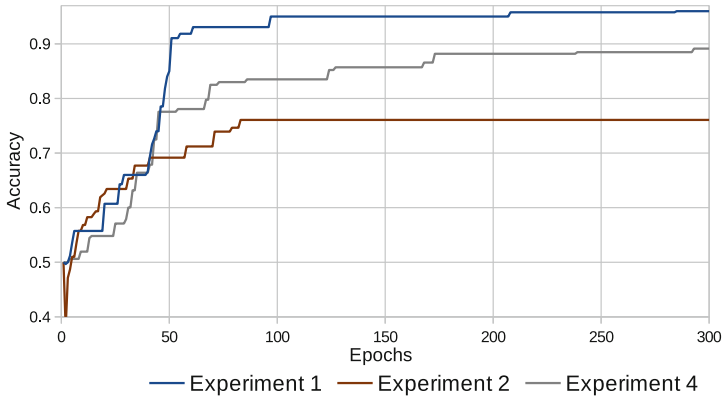


Fig. 3. Accuracy evolution for experiments 1, 2 and 4.

make the initial premise of putting the shapelet in evidence compromised, as the shapelet could be almost the entire length of the series. It becomes evident that setting parameters like the size of the extracted shapelets, the quantity of the extracted shapelets and the relevance of the shapelet is a difficult task.

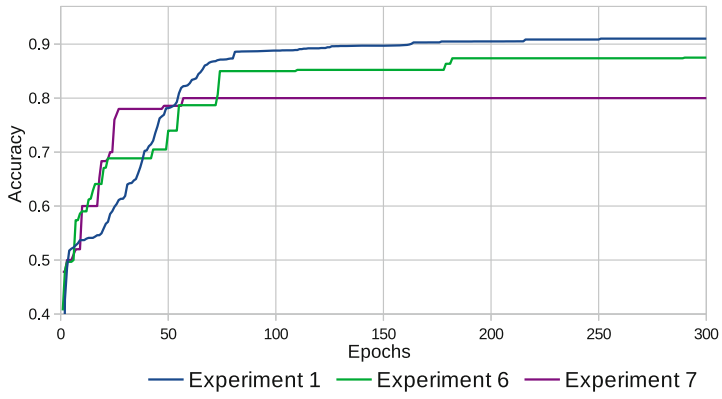


Fig. 4. Accuracy evolution for experiments 1, 6 and 7.

Another indicator of shapelet quality are the two cross-validation experiments 6 and 7. Both are inferior to experiment 1 at all epochal amounts (Fig. 4). Even so, experiment 6 was superior to experiment 7.

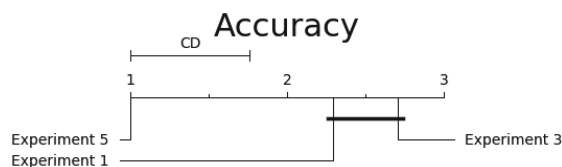


Fig. 5. Critical difference diagram for Nemenyi test.

Experiments 3 and 5 with transfer learn from the shapelet and noise series constitute the final objective. The result of the accuracy in the validation basis for experiments 1, 3 and 5 is described in Table 2 for each series (Fig. 6). For comparison, other 4 related classifiers in evidence were chosen, as described before. Table 2 is expressed as means (of 30 measurements), each series of measurements has a standard variation in the order of 10^{-5} . The Student’s t-test of both between experiments 1 vs 3 and separately experiments 1 vs 5 obtained p-values in the order of 10^{-15} each, so it is a high degree of reliability that the series present statistically relevant differences along the training epochs. After rejecting the null hypothesis, we then proceed with a post-hoc analysis in the best classifier as possible to see in diagram Fig. 5. The significance level of the tests is $\alpha=0.050$ and critical distance of 0.76. We can see that on final result experiment 3 and experiment 1 have no significant differences and experiment 5 is significant different.

An important result is that of these 20 sets, at least 15 sets (in experiments 3 and 5) were better when compared to experiment 1 (baseline). Another relevant factor is that the accuracy of the classification overtaken what is equivalent to the state-of-the-art in half of the cases.

When evaluating the state-of-the-art on the theme of symbolic representations, it was expected that the model trained with the shapelets series would be similar to the training model of the original series started from scratch. However, there were situations where transfer learning showed superior results when started from shapelets and in some other cases where the final result was better when training started by the noise set. In these, it seems that there is an external knowledge towards symbolic representations that support a more precise classification with less overfitting.

In many cases, the results obtained in terms of accuracy of the time series classifier reach the reference works. The same results in simple base training, using the implementation of [19], would be repeatedly similar. However, by including the symbolic representations (which, in most cases, are the classification criteria of domain experts), the weights adjust more quickly (which is expected) and improve the generalization of learning, two relevant benefits.

Table 2. Accuracy of the main methods with the best result in bold.

#	Datasets	Exp. 1	Exp. 3	Exp. 5	LTS	ELIS	BOSS	HIVE-COTE
1	BeetleFly	0.9500	0.9500	1.0000	1.0000	0.8500	0.9490	0.9590
2	BirdChicken	1.0000	1.0000	1.0000	1.0000	0.9000	0.9840	0.9505
3	ECGFiveDays	0.9978	0.9954	1.0000	0.9954	1.0000	0.9830	0.9895
4	ECG200	0.9100	0.9300	0.9000	0.9200	–	0.8900	0.8819
5	CBF	0.9933	0.9956	0.9978	0.9967	–	0.9980	0.9994
6	FaceFour	0.9318	0.9091	0.9659	0.9432	0.9545	0.9960	0.9495
7	FacesUCR	0.9268	0.9273	0.9210	0.9434	–	0.9510	0.9836
8	Gun_Point	1.0000	1.0000	1.0000	1.0000	0.9333	0.9940	0.9967
9	ItalyPowerDemand	0.9718	0.9974	0.9689	–	0.9757	0.8660	0.9678
10	Lightning7	0.8904	0.8493	0.8630	0.9178	0.8082	0.8100	0.8111
11	Lightning2	0.7869	0.7869	0.8197	0.7869	–	0.6660	0.7970
12	MoteStrain	0.9393	0.9377	0.9090	0.9361	0.8978	0.8460	0.9468
13	OliveOil	0.8000	0.7000	0.6667	0.9667	–	0.8700	0.8977
14	DiatomSizeRed	0.6536	0.5850	0.5850	–	0.8987	0.9390	0.9419
15	Coffee	1.0000	1.0000	1.0000	1.0000	0.9643	0.9890	0.9982
16	Symbols	0.9598	0.9545	0.9618	0.9889	0.7829	0.9610	0.9650
17	Beef	0.6333	0.7000	0.7000	0.9330	0.6333	0.6150	0.7227
18	SyntheticControl	0.9933	0.9933	0.9900	–	–	0.9680	0.9996
19	Trace	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
20	TwoLeadECG	1.0000	1.0000	1.0000	1.0000	0.9982	0.9850	0.9935

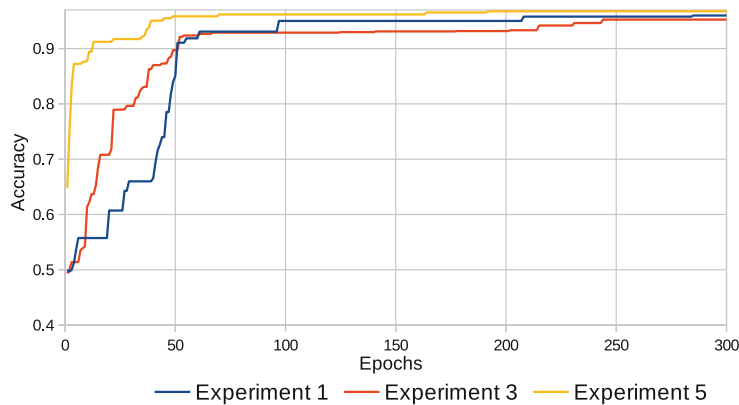


Fig. 6. Accuracy evolution for experiments 1, 3 and 5.

Figures 3, 4, 6, 7 and 8 show the accuracy evolution graphs of all experiments in comparison with experiment 1 as baseline.

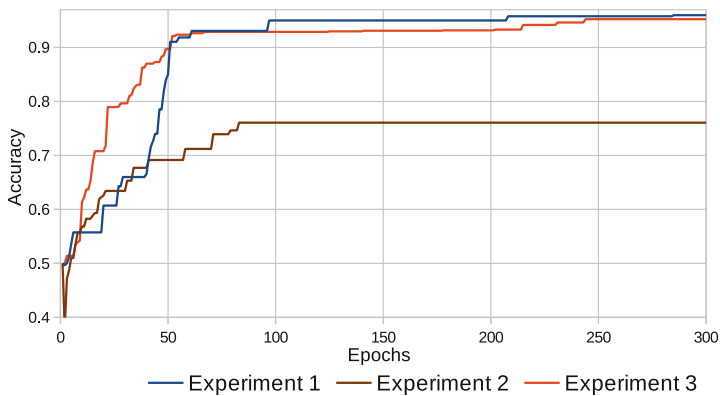


Fig. 7. Accuracy evolution for experiments 1, 2 and 3.

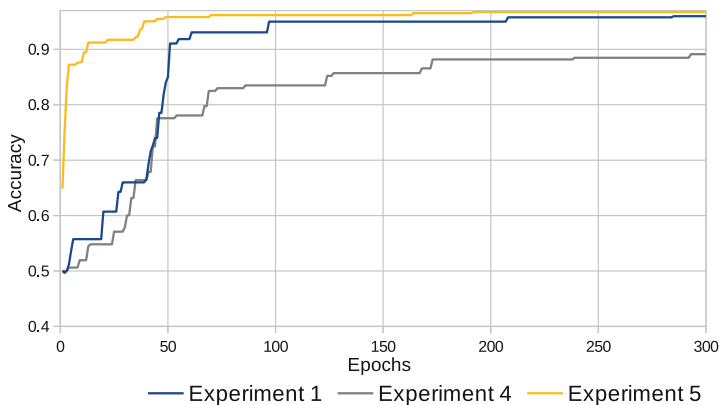


Fig. 8. Accuracy evolution for experiments 1, 4 and 5.

5 Conclusion

By exploring shapelets we obtained important results for time series classification in most of the tested databases. We believe that the objectives of this work have been achieved, since the performed experiments presented better results than training purely started from scratch. Thus, the importance of extracting symbolic representations in time series classifiers is evident, as they can be useful even in classifiers based on convolutional networks.

Learning based on neural networks is not trivial, as it requires the data analyst to adjust their experiments with different parameters. Only the continued training of the model, without parameter changes, would invariably lead to an increase in error. The proposed method, therefore, has a great advantage in this type of classifier.

Throughout this work, some points identified as limitations have already been pointed. Among them are experiments, whose results were less than expected, such as specific domains. Furthermore, there seems to be a need to resolve a question not yet fully answered: are there non-symbolic components relevant to classification that can be captured by a neural network? The quality of the extracted shapelets still needs to be separated from the external factors in every symbolic representation. By pointing out the limitations of this work, it is evident that derived studies are still needed. The results from the experiments allow us to believe that the quality of the extracted shapelets can be improved, making the proposed method more refined. Furthermore, due to the number of parameters, the tests need to be replicated in order to confirm that the results persist in different contexts, such as other neural network architectures.

The analysis of the model already trained in a convolutional neural network is also a necessary development. Given the complexity of an already adjusted set of weights, unsupervised learning can be applied to segment the concepts in the learned model, such as identifying symbolic representations in the activated weights of a neural network. This would give a non-experimental, more analytical perspective on what happens during the transfer learn process. This can lead to the suppression of the training stage of the shapelet series using a neural network and creating a new model from scratch.

Another interesting future work would be comparison with [9], the authors search for a shapelet using neural network architecture. The implementation presented in this paper extracts shapelets using neural network adjustment directly. According to authors this hybrid approach have similar results than extracting shapelets using traditional extraction. Likely mutual contributions are possible to get better results.

References

1. Bagnall, A., Lines, J., Hills, J., Bostrom, A.: Time-series classification with cote: the collective of transformation-based ensembles. *IEEE Trans. Knowl. Data Eng.* **27**(9), 2522–2535 (2015)
2. Cui, Z., Chen, W., Chen, Y.: Multi-scale convolutional neural networks for time series classification. *CoRR abs/1603.06995* (2016). <http://arxiv.org/abs/1603.06995>
3. Dau, H.A., et al.: The UCR time series classification archive (October 2018). https://www.cs.ucr.edu/~eamonn/time_series_data_2018/
4. Fang, Z., Wang, P., Wang, W.: Efficient learning interpretable shapelets for accurate time series classification. In: 2018 IEEE 34th International Conference on Data Engineering (ICDE), pp. 497–508 (April 2018). <https://doi.org/10.1109/ICDE.2018.00052>
5. Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L., Muller, P.: Transfer learning for time series classification. *CoRR abs/1811.01533* (2018). <http://arxiv.org/abs/1811.01533>
6. Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., Muller, P.-A.: Deep learning for time series classification: a review. *Data Min. Knowl. Disc.* **33**(4), 917–963 (2019). <https://doi.org/10.1007/s10618-019-00619-1>

7. Gamboa, J.C.B.: Deep learning for time-series analysis. arXiv preprint [arXiv:1701.01887](https://arxiv.org/abs/1701.01887) (2017)
8. 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, KDD 14, pp. 392–401. ACM (2014). <https://doi.org/10.1145/2623330.2623613>, <http://doi.acm.org/10.1145/2623330.2623613>
9. Guijo-Rubio, D., Gutiérrez, P.A., Tavenard, R., Bagnall, A.: A hybrid approach to time series classification with shapelets. In: Yin, H., Camacho, D., Tino, P., Tallón-Ballesteros, A.J., Menezes, R., Allmendinger, R. (eds.) IDEAL 2019. LNCS, vol. 11871, pp. 137–144. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33607-3_16
10. Långkvist, M., Karlsson, L., Loutfi, A.: A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recogn. Lett.* **42**, 11–24 (2014)
11. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**(7553), 436–444 (2015)
12. Li, G., Choi, B., Bhowmick, S.S., Wong, G.L.H., Chun, K.P., Li, S.: Visualet: visualizing shapelets for time series classification. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 3429–3432 (2020)
13. 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 Trans. Knowl. Eng.* (2020)
14. Lin, J., Keogh, E., Wei, L., Lonardi, S.: Experiencing sax: a novel symbolic representation of time series. *Data Min. Knowl. Disc.* **15**(2), 107–144 (2007)
15. Lines, J., Bagnall, A.: Time series classification with ensembles of elastic distance measures. *Data Min. Knowl. Disc.* **29**(3), 565–592 (2014). <https://doi.org/10.1007/s10618-014-0361-2>
16. Lines, J., Taylor, S., Bagnall, A.: Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles. *ACM Trans. Knowl. Discov. Data* **12**(5), 52:1–52:35 (July 2018). <https://doi.org/10.1145/3182382>, <http://doi.acm.org/10.1145/3182382>
17. Schäfer, P.: The boss is concerned with time series classification in the presence of noise. *Data Min. Knowl. Disc.* **29**(6), 1505–1530 (2015)
18. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**(1), 1929–1958 (2014)
19. Wang, Z., Yan, W., Oates, T.: Time series classification from scratch with deep neural networks: A strong baseline. *CoRR abs/1611.06455* (2016). <http://arxiv.org/abs/1611.06455>
20. Ye, L., Keogh, E.: Time series shapelets: a new primitive for data mining. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2009, pp. 947–956. ACM, New York (2009). <https://doi.org/10.1145/1557019.1557122>, <http://doi.acm.org/10.1145/1557019.1557122>
21. 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)