A Dictionary-Based with Stacked Ensemble Learning to Time Series Classification*

Rauzan Sumara[1][0000−0001−7643−4735], Wladyslaw Homenda[1,2][0000−0001−7787−4927], Witold Pedrycz[3][0000−0002−9335−9930], and Fusheng Yu[4][0000−0001−9144−9150]

[1] The Faculty of Mathematics and Information Science, Warsaw University of Technology, Warsaw, Poland
{rauzan.sumara.dokt,wladyslaw.homenda}@pw.edu.pl

[2] The Faculty of Applied Information Technology, University of Information Technology and Management, Rzeszow, Poland

[3] The Department of Electrical and Computer Engineering, University of Alberta, Edmonton, Canada
wpedrycz@ualberta.ca

[4] The School of Mathematical Sciences, Beijing Normal University, Beijing, China
yufusheng@bnu.edu.cn

Abstract. Dictionary-based methods are one of the strategies that have grown in the realm of time series classification. Particularly, these methods are effective for time series data that have different lengths. Our contribution involves introducing the integration of a dictionary-based technique with stacked ensemble learning. This study is unique since it combines the symbolic aggregate approximation (SAX) with stacking gated recurrent units (GRU) and a convolutional neural network (CNN), referred to as SGCNN, which has not been previously investigated in time series classification. Our approach uses the SAX technique to transform unprocessed numerical data into a symbolic representation. Next, the classification process is done using the SGCNN classifier. Empirical experiments demonstrate that our approach performs admirably across various datasets. In particular, our method achieves the second position among current advanced dictionary-based methods.

Keywords: Dictionary-based method · Stacked ensemble learning · Time series · Classification.

1 Introduction

The field of time series classification has garnered significant interest over the past decade. Within this domain, the prominence of dictionary-based classifiers has grown notably, finding application in diverse contexts. Recent investigations into neural networks in the field of natural language processing (NLP) served as the inspiration for this research.

* This research was supported by the Warsaw University of Technology within the Excellence Initiative: Research University (IDUB) programme.

We introduce a novel time series classification framework, commencing with transforming time series into a sequence of words using the SAX and employing stacked ensemble learning for classification. Our objective is to explore the efficacy of a stacking the GRU and CNN, called SGCNN, as a classifier for time series classification. A distinctive aspect of this study is the original integration of SAX and SGCNN, a combination previously unexplored in time series classification. The rationale behind employing ensemble learning lies in its recognized capability to enhance predictive accuracy when contrasted with individual models.

We consider transforming numeric data into a symbolic representation and generating sequences of words. This new form of the converted time series is then used to train the SGCNN model. The empirical experiment was conducted on 30 benchmark datasets from www.timeseriesclassification.com. Comparative analysis contrasts the classification results obtained by our method with those achieved by state-of-the-art approaches. We also provide the code to be publicly available through the link [5].

2 Literature Review

Various groups of algorithms are dedicated to the task of time series classification, with dictionary-based methods being particularly relevant to the scope of this study. In this context, a dictionary contains segments of time series represented as symbols, where each segment is construed as a word. Several dictionary-based algorithms have been introduced. For instance, the Bag of Patterns (BOP) algorithm stands out as a famous dictionary-based approach [1]. This algorithm employs time series data through a histogram representation, illustrating similarities to the well-known bag of words technique utilized in NLP. An extension of the BOP method is the SAX and Vector Space Model (SAXVSM) [2]. This method involves the generation of random subsequences from a time series, followed by the extraction of features for each subsequence. Another example from this category is dynamic time warping features (DTW-F) [3].

One widely used technique that relies on a dictionary is the bag of symbolic Fourier approximation symbols (BOSS) method [4]. The method employs symbolic Fourier approximation (SFA), which contributes to its notable resistance to noise. Another variation of this methodology, called contract BOSS (cBOSS) [5], has also been presented. The alteration that has been implemented pertains to the selection of parameters, which is an inherent part of the cBOSS algorithm and enhances the speed of the procedure. An additional classifier within the domain of dictionary-based methods is word extraction for time series classification with dilation (WEASEL-Dilation) [6]. Representing a novel iteration of a prior version, WEASEL, WEASEL-Dilation seeks to mitigate the issue of the considerable memory footprint associated with WEASEL. This is achieved by managing the search space through a randomly parameterized SFA transformation.

[5] github.com/rauzansumara/dictionary-based-with-stacked-ensemble-learning

DOI: 10.1007/978-3-031-63759-9_15

3 The Method

The proposed methodology can be deconstructed into four distinct stages. The initial stage involves standardizing the time series. The second step focuses on transforming the time series into a symbolic representation. Afterward, the third step involves creating word sequences and adding them to a corpus. The final step utilizes the generated corpus to train the SGCNN model.

Standardizing the time series The initial phase of the process involves preprocessing the data through z-standardization. This standardization process is accomplished by applying the z-score formulation $z_t = \frac{x_t - \mu_x}{\sqrt{\sigma_x^2}}$, with $t = 1, 2, \ldots, n$, where z_t denotes the standardized time series for the t-th element, x_t represents the observed value of the time series at the t-th time point, μ_x is the mean of the time series, and σ_x denotes the standard deviation of the time series.

Converting numeric time series into symbolic time series In the subsequent step of the technique, we used the simple SAX procedure as a potential approach for our study, where a time series will have the same length as the sequence of symbols. The size of the alphabets or symbols is represented by a. The discretization process involves equally dividing the area under the Gaussian curve into distinct intervals, denoted as $Ir = \{Ir_0, Ir_1, Ir_2, \ldots, Ir_{a-1}, Ir_a\}$. The first interval, Ir_0, represents negative infinity, while the last interval, Ir_a, represents positive infinity. It is important to note that each interval, Ir_j, is smaller than the subsequent interval, Ir_{j+1}, where j is less than a. If the time point falls inside a specified interval, the numerical time point is substituted with the designated alphabetical symbol. Using three alphabets, Fig. 1 demonstrates the example process of transforming numerical time series into symbols.

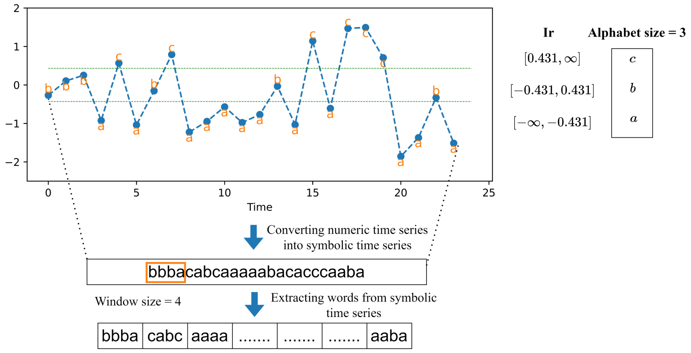


Fig. 1: The concept of transforming numerical data into symbols and generating a sequence of words.

Generating words from symbolic time series The window size (w) must be selected to extract words. The selection of the word length range is at the researchers' discretion. Therefore, we chose the condition $2 \leq w \leq 8$ because the average English word often consists of two to eight characters. If n represents the length of the symbolic time series, then a word list consisting of n/w words is generated for each symbolic time series. The method of extracting words is illustrated in Fig. 1 with a window size of 4.

Training stacked ensemble learning model In this paper, we introduced the SGCNN model, which is a combination of GRU positioned at the top and followed by CNN layers. It is a modified architecture from stacked ensemble learning introduced in the NLP paper [7]. Modification of this architecture, presented in Fig. 2, revolved around removing several layers and trying with different building blocks to make sure that it is not prone to either overfitting or underfitting. Firstly, a 4-dim embedding layer is acquired through training in order to align with the surrounding context of each word. Embeddings are then passed through GRU layers. In the GRU, the update gate (u_t) enables the model to determine the information from the previous hidden states (H_{t-1}) used to describe the future. Moreover, the reset gate (r_t) works to determine the information of H_{t-1} to be forgotten, and it is used to construct the current memory (c_t) for capturing the relevant information in H_t, where $u_t = \sigma\left(W_u z_t + U_u H_{t-1}\right)$, $r_t = \sigma\left(W_r z_t + U_r H_{t-1}\right)$, $c_t = tanh\left(W_c z_t + r_t \odot U_c H_{t-1}\right)$, and $H_t = (1 - z_t) \odot H_{t-1} + z_t \odot c_t$. These components rely on learning parameter matrices W, U, and a sigmoid activation function $\sigma(\dots)$.

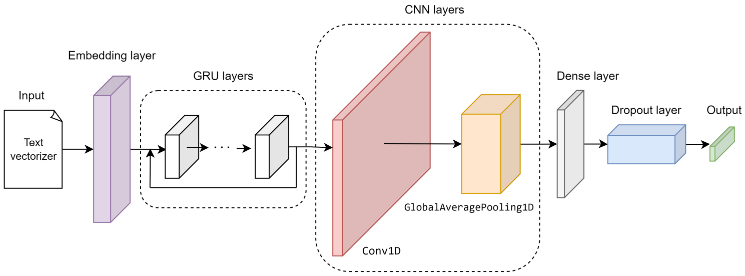


Fig. 2: Proposed stacked deep learning architecture.

Next, 16 units of the GRU layer process this input to extract features for subsequent layers followed by a CNN layer that incorporates the output generated by the GRU model. A 1 × 1 kernel window size and 16 filters are utilized in the CNN layer. Another important type of layer is a global average pooling (GAP). Following that, there is a dense layer with 32 neurons, and a dropout layer with a rate of 0.1 is utilized to decrease the complexity of SGCNN. Finally, this architecture uses a softmax activation function for the final layer.

4 Results

The study conducted experiments using a set of time series datasets that are consisted of thirty univariate time series, as shown in Table 1. All required calculations were performed in Python programming language. We fitted the models with different hyperparameters, such as using alphabet size $a = \{5, 7, 8, 9, 11, 12, 13, 15, 17, 18, 19, 20\}$ and words of length $w = \{2, 3, 5, 7, 8\}$. We constructed the model using an Adam optimizer with a loss function of a categorical cross-entropy, 100 epochs, 0.001 learning rate and 16-batch size. During the evaluation, the compared outcomes correspond only to the test sets. Last but not least, we did a post-hoc test using the Wilcoxon-Holm with a significance level (α) of 5%.

Heatmap plots for a few chosen datasets are displayed in Fig. 3, allowing us to examine the effects of alphabet size and word length on classification accuracy. The figures illustrate the variations in accuracy that occur when we alter the word length w and alphabet size a. To achieve optimal outcomes, various datasets are frequently attained while utilizing alphabet sizes ranging from 11 to 20 and word lengths of 7 to 8. Table 1 corresponds to the best accuracy achieved by our approach and other dictionary-based methods. Apart from this, we also consider several competitors under a group of non-dictionary-based methods, such as TSF [8], RISE, and STC [9], proximityforest [10], InceptionTime [11], Catch22 [12], TS-CHIEF [13], HIVE COTE or HC1 [14], ROCKET [15], and RSTSF [16]. Overall, the proposed method performs very competitively compared to other classification methods. The fact that our model is parameterized contributes to this outcome, where we evaluate and select the optimal model for a specific dataset using the best hyperparameter configurations. As a result, our approach surpasses most dictionary-based methods in terms of accuracy except WEASEL-Dilation. Our method also performs at comparable levels or even better than several non-dictionary-based methods. A critical difference diagram is shown in Fig. 4, which illustrates the average ranks of each method on 30 datasets. A lower mean rank implies the better accuracy of a particular method,

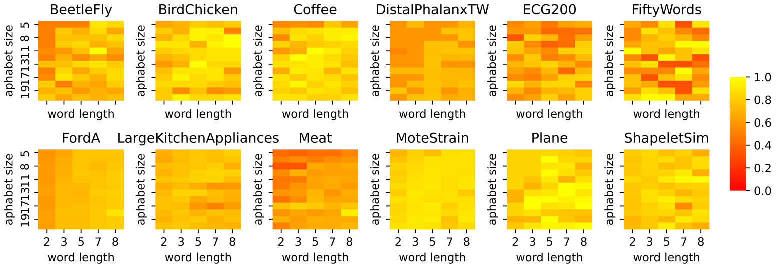


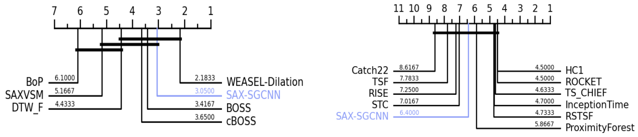
Fig. 3: Test accuracy for selected datasets based on alphabet size and word length.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_15

Table 1: Our method compared to two groups of methods in terms of accuracy (Acc in %)

Datasets	Dictionary-based		Non dictionary-based		Proposed		
	Acc	Algoritm	Acc	Algoritm	Acc	a	ω
BeetleFly	95.00	BOSS	**100.00**	TS_CHIEF	**100.00**	11	7
BirdChicken	**100.00**	BOSS	95.00	InceptionTime	**100.00**	9	8
Coffee	**100.00**	BOSS	**100.00**	TSF	**100.00**	12	3
CricketZ	80.00	WEASEL-D	**85.90**	ROCKET	78.70	11	7
DistalPhalanxOutlnCor.	78.62	WEASEL-D	**80.43**	ProximityForest	77.90	11	7
DistalPhalanxTW	70.50	WEASEL-D	70.50	HC1	**77.54**	11	8
Earthquakes	74.82	SAXVSM	75.54	ProximityForest	**83.69**	11	8
ECG200	89.00	WEASEL-D	**92.00**	ROCKET	76.81	8	8
FaceFour	**100.00**	BOSS	**100.00**	TS_CHIEF	95.45	15	3
FiftyWords	83.30	WEASEL-D	84.84	TS_CHIEF	**98.90**	11	2
FordA	95.61	WEASEL-D	**98.03**	RSTSF	89.74	18	8
GunPoint	**100.00**	BOSS	**100.00**	STC	**100.00**	8	8
Herring	65.63	WEASEL-D	70.31	InceptionTime	**70.94**	17	8
InlineSkate	47.45	cBOSS	**67.45**	RSTSF	42.02	15	7
ItalyPowerDemand	96.02	DTW_F	**97.08**	RSTSF	74.67	9	7
LargeKitchenAppliances	87.73	SAXVSM	**90.40**	InceptionTime	84.27	9	7
Lightning7	76.71	WEASEL-D	**84.93**	ProximityForest	73.97	13	8
Meat	91.67	WEASEL-D	**96.67**	RSTSF	92.67	18	8
Medicallmages	78.42	WEASEL-D	**82.63**	RSTSF	72.76	20	8
MiddlePhalanxOutlnCor.	**84.54**	WEASEL-D	84.19	HC1	75.53	5	5
MiddlePhalanxTW	55.84	cBOSS	59.74	RISE	**62.02**	15	7
MoteStrain	95.21	WEASEL-D	**95.85**	HC1	92.33	19	5
OliveOil	**96.67**	WEASEL-D	93.33	RISE	91.93	17	5
Plane	**100.00**	BOSS	**100.00**	TSF	**100.00**	20	8
ProximalPhalanxOutlnCor.	91.07	WEASEL-D	**93.13**	InceptionTime	83.85	12	8
ProximalPhalanxTW	81.95	BOSS	**82.44**	TS_CHIEF	72.60	9	7
ShapeletSim	**100.00**	BOSS	**100.00**	STC	98.89	11	8
SyntheticControl	99.67	WEASEL-D	**100.00**	ROCKET	62.33	5	5
Trace	**100.00**	BOSS	**100.00**	ProximityForest	88.00	11	7
Worms	68.83	WEASEL-D	**81.82**	TS_CHIEF	62.34	19	5

while a solid horizontal bar implies statistical insignificance between two or more methods. According to average ranks, our approach, SAX-SGCNN, is slightly better than BOSS and cBOSS, even though it is significantly insignificant. The most accurate dictionary method is WEASEL-Dilation, which is in the same group as other dictionary-based methods. However, the horizontal bar connects to our approach, which means they are statistically indifferent. The similarity in behavior observed among SAX-SGCNN, BOSS, cBOSS, and WEASEL-Dilation can be attributed to the similarity in their respective approaches. All of them convert the time series into a symbolic representation and then analyze the sequence of words. Moreover, the most accurate non-dictionary-based methods



(a) (b)

Fig. 4: Plot of critical difference based on average ranks from 30 datasets on test accuracy in comparison with (a) the dictionary-based and (b) the non-dictionary-based techniques.

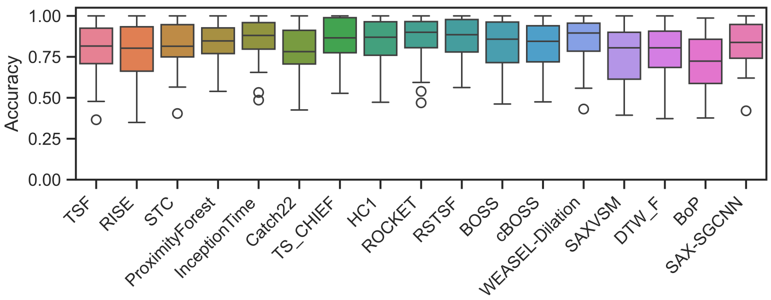


Fig. 5: A box plot representing the accuracy of test set from 30 datasets.

such as HC1, ROCKET, TS-CHIEF, InceptionTime, RSTSF, and ProximityForest are statistically insignificant in comparison to our approach. The box plots in Fig. 5 show how close the accuracy of our approach to the current state-of-the-art methods. Top-ranking approaches have a low interquartile range (IQR), few outliers, and a median accuracy above 80%. Based on the results obtained, the proposed approach performs relatively better than the others in many cases.

5 Conclusion

The time series classification method that we describe in this paper works well with various datasets. Among dictionary-based methods, our approach, SAX-SGCNN, outperforms all other methods except WEASEL-Dilation. One challenge of the presented method is regarding hyperparameter tunings, such as the number of alphabets and the word length, which must be tuned for each dataset separately to obtain the best results. Future work related to this research can be focused on extending this approach to classify the multivariate time series data, and choosing a better way of arranging hyperparameters to eliminate the problem will also be the subject of further modifications.

References

1. Hatami, N., Gavet, Y., Debayle, J.: Bag of recurrence patterns representation for time-series classification. Pattern Anal Applic. 22, 877–887 (2019).
2. Senin, P., Malinchik, S.: SAX-VSM: Interpretable Time Series Classification Using SAX and Vector Space Model. In: 2013 IEEE 13th International Conference on Data Mining. pp. 1175–1180. IEEE, Dallas, TX, USA (2013).
3. Franses, P.H., Wiemann, T.: Intertemporal Similarity of Economic Time Series: An Application of Dynamic Time Warping. Comput Econ. 56, 59–75 (2020).
4. Schäfer, P.: The BOSS is concerned with time series classification in the presence of noise. Data Min Knowl Disc. 29, 1505–1530 (2015).
5. Middlehurst, M., Vickers, W., Bagnall, A.: Scalable Dictionary Classifiers for Time Series Classification. In: Yin, H., Camacho, D., Tino, P., Tallón-Ballesteros, A.J., Menezes, R., and Allmendinger, R. (eds.) Intelligent Data Engineering and Automated Learning – IDEAL 2019. pp. 11–19. Springer International Publishing, Cham (2019).
6. Schäfer, P., Leser, U.: WEASEL 2.0 – A Random Dilated Dictionary Transform for Fast, Accurate and Memory Constrained Time Series Classification. (2023).
7. Lee, E., Rustam, F., Washington, P.B., Barakaz, F.E., Aljedaani, W., Ashraf, I.: Racism Detection by Analyzing Differential Opinions Through Sentiment Analysis of Tweets Using Stacked Ensemble GCR-NN Model. IEEE Access. 10, 9717–9728 (2022).
8. Tan, C.W., Bergmeir, C., Petitjean, F., Webb, G.I.: Time series extrinsic regression: Predicting numeric values from time series data. Data Min Knowl Disc. 35, 1032–1060 (2021).
9. Shu, W., Yao, Y., Lyu, S., Li, J., Chen, H.: Short isometric shapelet transform for binary time series classification. Knowl Inf Syst. 63, 2023–2051 (2021).
10. Lucas, B., Shifaz, A., Pelletier, C., O'Neill, L., Zaidi, N., Goethals, B., Petitjean, F., Webb, G.I.: Proximity Forest: an effective and scalable distance-based classifier for time series. Data Min Knowl Disc. 33, 607–635 (2019).
11. 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 Min Knowl Disc. 34, 1936–1962 (2020).
12. Lubba, C.H., Sethi, S.S., Knaute, P., Schultz, S.R., Fulcher, B.D., Jones, N.S.: catch22: CAnonical Time-series CHaracteristics: Selected through highly comparative time-series analysis. Data Min Knowl Disc. 33, 1821–1852 (2019).
13. Shifaz, A., Pelletier, C., Petitjean, F., Webb, G.I.: TS-CHIEF: a scalable and accurate forest algorithm for time series classification. Data Min Knowl Disc. 34, 742–775 (2020).
14. Lines, J., Taylor, S., Bagnall, A.: HIVE-COTE: The Hierarchical Vote Collective of Transformation-Based Ensembles for Time Series Classification. In: 2016 IEEE 16th International Conference on Data Mining (ICDM). pp. 1041–1046. IEEE, Barcelona, Spain (2016).
15. Dempster, A., Petitjean, F., Webb, G.I.: ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. Data Min Knowl Disc. 34, 1454–1495 (2020).
16. Cabello, N., Naghizade, E., Qi, J., Kulik, L.: Fast, Accurate and Interpretable Time Series Classification Through Randomization. (2021).