”ROCKET Exceptionally fast and accurate time series classication using random convolutional kernels” - Paper Review

Elad Eatah

Paper Overview

The paper [1] was published on arXiv on October 2019. This work suggests using a collection of random convolution kernels as a features-transform for time-series. This new features are then utilized as inputs for classical classification models, such as logistic-regression and ridge-regression.

Many existing state-of-the-art solutions simply cannot scale for larger datasets. Some of them are quite slow for small datasets, or even intractible for larger datasets. In addition, some methods incorporate very few specific features, such as shape features of frequency features. In this method, all parameters are drawn randomly. Thus there is no “training stage” for this transform, as in many other SOTA solutions. Consequently, this method uses only a small fraction of the computation-time used in other methods (including their training-time), while presenting state-of-the-art accuracy, and rivaling the accuracy of many existing alternatives. The researchers named this tranform **ROCKET** - **R**and**O**m **C**onvolutional **KE**rnel **T**ransform.

The researchers then demonstrate the performance of their method by applying this feature-map on various datasets of time-series with varying size, and using the output features for classification tasks via ridge-regression and logistic-regression.

Roughly a year later, two of the the researchers published the paper [2], which presents an improved algorithm named **MiniROCKET** (for **MINI**mally ROCKET). This paper demostrastes that by removing most of the randomness of ROCKET, the computation can be made mcuh faster (even 75 times faster for some larger datasets) than ROCKET, whereas the state-of-the-art accuracy is essentialy maintained.

This review deals mainly with ROCKET, but MiniROCKET is mentioned a few times for comparison.

Related SOTA methods

The method of *Proximity-Forest* (presented in [5]) is an example of a scalable classifier which attains state-of-the-art accuracy. This ensemble model consists of decision-trees, and their splitting-criterion is based on the elastic distance measure. This method was the basis for the *TS-CHIEF* method [6], whose splitting-criterion is both dictionary-based and interval-based. Both method have training-time complexity which is quasilinear in the number of sequences training samples n and quadratic in the length of the input sequences l.

The intuition for using convolution-kernels can be found in the paper [4]. This paper presents the *InceptionTime* model, an ensemble classifier for time-series. This model contains 5 CNNs which incorporate the concept of inception-blocks. The authors of this paper speculates, and later demonstrates, that since time-series and images are practically 'the same' (up to reshape), the tremendous success of the inception architecture for images-classification mau be carried-away to time-series. The training of this model is performed using stochastic gradient-descent (*SGD*, or any of its variants, such as *Adam*). Hence its training-time is essentialy-linear in n and can be parallelized using multi-GPUs.

The idea of random convolution-kernels, as a features-transform for time-series, can be found in [3]. In this work, random convolution-kernels are lernt using unsupervised-learning. Their model is a multi-layer convolutional architecture, and the dilation in the convolutions of each layer is exponentially-larger than the dilation of the previous layer. The lernt features are then used as an input for a support-vector machine.

ROCKET Description

Simply said, the algorithm ROCKET is a set of convolution-kernels, i.e a single 1D convolutional-layer. Let k be the number of these kernels. Each of these kernels is fully-characterized by the following attributes: kernel length l_k, padding, kernel weights w, bias b, dilation d and stride, which is always 1. Each of the kernels is applied, as a sliding dot-product, to each input time-series X. Formally, the convolution is performed, from the i-th position of the input series using the following formula

$$X_i * w = \sum_{j=0}^{l_k-1} X_{i+jd} w_j, \tag{1}$$

where X_i is the i-th element of X. From each convolution output, two features are computed:

1. The maximum value of the output (in a way, a global max-pooling of the output).
2. The proportion of positive-values (*ppv*) in the output sequence.

The concept of global max-pooling is known for dimensionality-reduction and spatial-invariance for images (for time-series, the invariance is rather temporal). The ppv feature captures the the proportion of the input, which correlates with the pattern of this specific convolution kernel.

Thus, for k convolution-kernels, there are $2k$ features in the output of the transform for each input time-series.

The parameters of each convolution kernel are determined as follows:

- The decision on padding (or not) is drawn randomly with uniform probability. If padding is used, then the padding is two-sided, with length $\frac{1}{2}(l_k - 1)\,d$ from each end of the input series. This padding allows the kernel to be centered on each of the (original) input elements. Padded kernels may detect patterns at either ends of the input sequences better than non-padded kernels.
- Kernel length is drawn randomly with uniform distribution from the set $\{5, 7, 9\}$. Then the kernels are usually much shorter than the input time-series.
- Kernel weights are first drawn from standard normal distribution $\mathcal{N}(0, 1)$. Then these weights are mean-centered (where the mean is performed over all weights).
- The bias is drawn from (continuous) uniform distribution $\mathcal{U}(-1, 1)$. Since the ppv features capture only positive values, then using two kernels which are similar, except for their biases, can 'highlight' different aspects in the output features-map.

- The dilation d is sampled using $d := \lfloor 2^x \rfloor$ where $x \sim \mathcal{U}(0, A)$, $A := \log_2 \frac{l_{input}-1}{l_k-1}$ and l_{input} is the length of input time-series. The varying of dilation allows for finding the same patterns, is different frequencies and scales.

The input sequences for the output features-map are assumed to be zero-meaned and have standard-deviation of 1.

The development process of this method was performed over 40 randomly-selected datasets from the UCR archive, so this method cannot overfit the UCR archive entirely.

Advantages and Drawbacks of The Method

The greatest advantage of this method, over many of the existing solutions, stems from its greatest novelty - almost no time is spent on fitting the features transform. The "fit" performs solely the random choices of parameters, and the time required for them is negligble w.r.t both the number os time-series samples and the length of the input sequences. Hence, the computation of the entire features-map, which depends on the choice of the number of kernels k, can still be very quick, even for large datasets. The most time-consuming step in the entire classification process is the fitting and inference of the following classification model, i.e logistic-regression or ridge-regression. For these specific classifiers, this step is still much quicker than the fitting of some other state-of-the-art solution, such as InceptionTime [4], Proximity-Forest [5] or TS-CHIEF [6].

An additional important advantage in this method is its (relative) conceptual-simplicity. Unlike other methods, there are no complex network structures or long and complicated computations in this method, just random numbers generations from simple and well-known distributions. Thanks for such simplicity, one may understand the method, implement it and maintaining its code with ease.

Moreover, since the convolution computations for different kernels do not rely on each other, the ROCKET algorithm can easily be distributed to multi-CPU machines and clusters. Furthermore, the state-of-the-art performance can be achieved on server CPUs. In other words, no specialized hardware is required, while other methods, as many machine and deep-learning models, depend on GPUs or TPUs in order to reach state-of-the-art performance.

The randomness of this algorithm also leads to a drawback. As all parameters are selected at random, these parameters' values cannot be explained easily and any additional information regarding the data cannot be utilized. One may even (erroneously) believe that this method is nothing more than a pure random guess. However, as the paper [1, p. 5] states, even random convolution-kernels can be viewed as frequency-selective filters. Thus, despite all the randomness, the choice of parameters may still be informative, but deciphering their meaning requires more attention (one may use the time saved by not training the features-map to do so...). These issues are slightly mitigated in MiniROCKET, where only a few of the parameters are chosen randomly (the rest are selected deterministically). The paper [2] even suggests a variant of the algorithm, which is completely deterministic, at the expense of additional computational resources. As a result, the amount of randomness in the algorithm turns out to be significantly lower, and can even be controlled.

Performance Comparison

The paper demonstrates the performance of this method on many datasets, against some of the state-of-the-art methods, in terms of both classification accuracy and total run-time (training+inference, whenever training is required). The comparisons are performed over a cluster of server CPUs where only a single CPU core is utilized for each used CPU (no speedups of parallel computation).

The first experiment was conducted on 85 'bake off' datasets from the UCR archive. ROCKET was performed with $10,000$ kernels, and the presented run-times are the mean of 10 different runs (with different kernels on each run). ROCKET's performance is compared against the published results (over the entire archive) of the following state-of-the-art methods: Proximity-Forest [5], TS-CHIEF [6], BOSS, Shapelet-Transform, HIVE-COTE, ResNet and InceptionTime [4]. Over all 85 datasets, ROCKET attains the best mean-accuracy (the same occurs on the 40 'development' datasets, while on the remaining 45 datasets, ROCKET is only second to TS-CHIEF. Then a Wilcoxon signed-rank test was performed to determine if the differences in accuracy are statistically-siginificant. In all these cases, the differences between ROCKET and the best remaining methods are not statistically-significant.

These experiment was conducted again on additional 43 datasets, in order to show ROCKET can scale to different series length. Since they were no published results for the other methods on these datasets, and because adapting these methods to varying-length series is nontrivial, the comparison was made only to another state-of-the-art method 1NN-DTW. On 39 of these datasets, ROCKET outperformed 1NN-DTW, was substantially more accurate on most of them.

A second type of experiment was conducted to show the scalability of ROCKET to different training-set sizes. Specifically, up to approximately 1 million time-series samples from the 'satellite Image Time Series' dataset were used . Here ROCKET was applied with 100, 1000 and $10,000$ kernels separately. Their features-maps were used as inputs to a logistic-regression classifier (further details on the training procedure are available in the paper). These models were compared with TS-CHIEF and Proximity-Forest, in terms of both accuracy and training-time. In all cases and training set sizes, ROCKET was dramatically-faster then the rest of the methods - taking 1 hour at most, in comparison to at least 10 hours for TS-CHIEF and Proximity-Forest. As for the classification accuracy, ROCKET with $1,000$ and $10,000$ kernels achieve similar accuracy as TS-CHIEF and Proximity-Forest. When using only 100 kernels, ROCKET is bested by TS-CHIEF and Proximity-Forest, but it requires only abount a minute for training. To sum up, ROCKET seems much more scalable in comparison to the other state-of-the-art methods.

References

[1] Angus Dempster, François Petitjean, and Geoffrey I. Webb, *Rocket: Exceptionally fast and accurate time series classification using random convolutional kernels*, `https://arxiv.org/abs/1910.13051`, 2019.

[2] Angus Dempster, Daniel F. Schmidt, and Geoffrey I. Webb, *Minirocket: A very fast (almost) deterministic transform for time series classification*, `https://arxiv.org/pdf/2012.08791.pdf`, 2020.

[3] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi, *Unsupervised scalable representation learning for multivariate time series*, `https://arxiv.org/abs/1901.10738`, 2020.

[4] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F. Schmidt, Jonathan Weber, Geoffrey I. Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean, *Inceptiontime: Finding alexnet for time series classification*, Data Mining and Knowledge Discovery **34** (2020), no. 6, 1936–1962, `http://dx.doi.org/10.1007/s10618-020-00679-8`.

[5] Benjamin Lucas, Ahmed Shifaz, Charlotte Pelletier, Lachlan O'Neill, Nayyar Zaidi, Bart Goethals, François Petitjean, and Geoffrey I. Webb, *Proximity forest: an effective and scalable distance-based classifier for time series*, Data Mining and Knowledge Discovery **33** (2019), no. 3, 607–635, `http://dx.doi.org/10.1007/s10618-019-00617-3`.

[6] Ahmed Shifaz, Charlotte Pelletier, François Petitjean, and Geoffrey I. Webb, *Ts-chief: a scalable and accurate forest algorithm for time series classification*, Data Mining and Knowledge Discovery **34** (2020), no. 3, 742–775, `http://dx.doi.org/10.1007/s10618-020-00679-8`.