ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

Federated Time Series Classification with ROCKET features

Bruno Casella[1,†], Matthias Jakobs[2,†], Marco Aldinucci[1] and Sebastian Buschjäger[2] *

1 - Alpha Research Group, Computer Science Department
University of Turin, C.so Svizzera 185, Turin - Italy

2 - Lamarr Institute for Machine Learning and Artificial Intelligence
TU Dortmund University, Dortmund, Germany

† Both authors contributed equally to this work.

Abstract. This paper proposes FROCKS, a federated time series classification method using ROCKET features. Our approach dynamically adapts the models' features by selecting and exchanging the best-performing ROCKET kernels from a federation of clients. Specifically, the server gathers the best-performing kernels of the clients together with the associated model parameters, and it performs a weighted average if a kernel is best-performing for more than one client. We compare the proposed method with state-of-the-art approaches on the UCR archive binary classification datasets and show superior performance on most datasets.

1 Motivation

Time series classification (TSC) is a central challenge in diverse domains, including healthcare, sensor analysis in the Internet of Things (IoT), and movement recognition for human behavior analysis. While deep learning approaches have demonstrated unprecedented performance in some areas, they come at substantial computational costs. Moreover, in many cases, simpler algorithms such as K-Nearest Neighbor with Dynamic Time Warping, Shapelets, or the relatively recent ROCKET (Random Convolutional Kernel Transform) can outperform deep learning methods with much lower computational costs[1, 2].

ROCKET [3] has emerged as a particularly effective method for TSC due to its unique approach to feature transformation. It utilizes a set of randomly sampled convolutional kernels to transform data, which is subsequently processed by a highly regularized linear model to select significant features. This method not only reduces the model size but, due to the random sampling, is also well-suited for low-resource environments. However, ROCKET is a centralized algorithm that requires data aggregation at a single node. This is particularly restrictive in IoT contexts where continuous data transmission to a central repository is impractical due to bandwidth and energy constraints.

*This research has been partly funded by the Federal Ministry of Education and Research of Germany and the state of North Rhine-Westphalia as part of the Lamarr Institute for Machine Learning and Artificial Intelligence, and partly supported by the Spoke "FutureHPC & BigData" of the ICSC - Centro Nazionale di Ricerca in "High Performance Computing, Big Data and Quantum Computing", funded by European Union - NextGenerationEU, and by the Horizon2020 RIA EPI project (G.A. 826647).

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

To address these challenges, we introduce a novel algorithm called Federated ROCKet featureS (FROCKS). FROCKS combines the ROCKET feature transformation with federated learning so that each client learns its own set of random features for its local data distribution. Then, it periodically communicates the most important kernels to a central server, combining them into a global model containing the most important features across all clients. Finally, the global model is communicated back to each client. The difficulty of this novel approach lies in the fact that we cannot use the standard FedAvg [4] method to average the models because each client can potentially have a different set of kernels, leading to different data preprocessing. Hence, FROCKS merges models by extracting the most important kernels from each client (i.e., using the top-K) instead of the conventional FedAvg approach.

Our contributions can be summarized as follows: (1) We propose a naive adaptation of ROCKET to federated learning and demonstrate its limitations. (2) We introduce FROCKS, which effectively combines the strengths of ROCKET with federated learning. (3) We show the efficacy of FROCKS through extensive experiments across all the binary classification datasets of the UCR archive, showing that it surpasses traditional methods in terms of F_1 score on most of the datasets with minimal communication rounds.

2 Related Work and Methodology

ROCKET: In [3], the authors present a time series feature extraction method using random convolutional kernels called ROCKET. Instead of learning the kernel weights and hyperparameters to transform the time series, ROCKET opts for sampling thousands of kernels randomly. Each kernel length is uniformly chosen in $\{7, 9, 11\}$ with normal distributed kernel values and bias values uniformly in $[-1, 1]$. To account for different time scales, each kernel also samples a random dilation value. During inference, after applying each random kernel to the time series, the authors utilize two non-linear activation functions to aggregate each transformed output into two real values. One value is the maximum value of each kernel's output, while the other is the *percentage of positive values* (PPV), measuring the number of kernel outputs that exceed zero. We only utilize PPV in our work since no statistically significant benefit has emerged by choosing both the maximum and PPV over just using PPV alone.

Naive approach: As mentioned, our goal is to classify time series in a federated setting, i.e., we have given a set of C clients where each client has access to local data, but we can only communicate models (and not the entire local dataset) to a central server. The central server can combine the individual models and redistribute a global model to each client. In this setting, the arguably most straightforward approach to use ROCKET for time series classification is as follows: Before starting the federated training, the central server creates K ROCKET kernels that are broadcast to all the clients, together with an initial logistic regression model. The clients will transform their local data according to the received ROCKET kernels and then fit their local model on the trans-

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

formed data. Then, each client communicates its model to the central server, where FedAvg [4] is applied to average all models. Finally, the global model is communicated back to the local models, where training continues on the transformed features. Note that in this version, all models share the same feature transformation, i.e., the same ROCKET kernels.

FROCKS: In FROCKS, we propose that every client receives its own set of kernels, i.e., we apply different feature transformations at each client. To do so, the server initializes every client with a different set of kernels that is used to train a logistic regression. After training this model, each client collects the $p = \lfloor \frac{K}{C} \rfloor$ best-performing kernels in terms of absolute weight and sends these kernels with their corresponding weights to the central server. At the central server, we gather all kernels to build a new set of kernels. If two clients send the same kernel, the server averages the corresponding weights. Finally, at the beginning of the next round, the central set of weights and kernels is again distributed to each client, and the new data transformation is applied to the local data. We highlight some advantages of this approach: First, there will never be more than K kernels in total, limiting the overall communication costs. Second, we can identify each kernel by its unique random seed used to generate it instead of sending the actual kernel, further reducing communication costs. Third, we can detect the convergence of FROCKS by checking if the set of weights and kernels changes significantly from round to round at the central server. More specifically, if we do not detect a change in the kernels and a change in the weights[1] at the central server in two consecutive rounds, then the overall approach is converged, and we can stop training.

3 Experiments

The goal of our experiments is to study the predictive performance of FROCKS and compare it against the naive baseline and a logistic regression trained on the raw features in a federated setting. We use the following experimental protocol:

Testbed setup: The baseline experiments that do not use any rocket features but train a logistic regression via FedAvg (we call this 'raw data'), as well as the naive experiments that distribute rocket features and train a logistic regression via FedAvg (we call this 'naive'), have been executed in a real distributed environment encompassing one server and four clients, each deployed on a dedicated server with an Intel®Xeon®processor (Skylake, IBRS, 8 sockets of one core) and one Tesla T4 GPU. We adopted OpenFL [5] as the FL framework and PyTorch to train the models. FROCKS ran a simulated federation made of one server and four clients on a machine with the previously listed hardware specification[2]. We used Scikit-learn as the library for training the logistic regression.

[1] We use $|w_{r-1} - w_r| \leq 10^{-8} + 10^{-5} \cdot |w_r|$ to detect if there is a sufficient change between round r and $r-1$.

[2] Since we propose a novel averaging technique and we need to communicate kernels instead of model weights, we were not able to implement FROCKS directly in OpenFL without significant code refactoring of the framework. We are currently exploring options on how to implement FROCKS in OpenFL.

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

Datasets and Models: For benchmarking, we focus on the binary time series classification datasets of the UCR archive[6]. For comparability, we followed the parameters of the original ROCKET paper [3]. The baseline experiments, i.e., the naive approach with raw data and with ROCKET features, train a logistic regression model for 100 rounds, minimizing the cross-entropy loss. Adam was used as an optimizer, with a learning rate of 10^{-3}. The batch size was fixed to $\{2, 4, 8\}$ according to the sample size of the dataset. As we were mainly interested in model performance, we found that the best solver for FROCKS was the L-BFGS optimizer. We have run experiments with $K \in \{100, 1\,000, 10\,000\}$ ROCKET kernels. We randomly split the data into training and testing data in each experiment and distributed it to each client, where it is used for local model training and testing phases. The global model is finally tested on the test data. Note that the data distribution adheres to the principle of independent and identically distributed (IID), meaning that each client holds data representative of the overall dataset. All experiments were repeated five times with different random seeds, and we reported average results. Since some datasets are imbalanced, we will focus on the F_1 score in our analysis. Due to space constraints, we focus on the 'Image' and 'Motion' categories of the UCR Archive in this paper. Additional results, encompassing more datasets and metrics, such as the top-1 accuracy, as well as the code required to replicate our experiments, are available at `https://github.com/MatthiasJakobs/FROCKS`.

Dataset	Raw	Naive			FROCKS		
	data	100	1000	10000	100	1000	10000
Yoga	0.68	0.68	0.69	0.70	0.62	0.67	**0.80**
WormsTwoClass	0.49	0.49	0.49	0.49	0.55	0.73	**0.77**
ToeSegmentation2	0.20	0.20	0.20	0.20	0.50	0.63	**0.66**
ToeSegmentation1	0.49	0.49	0.49	0.49	0.67	0.78	**0.82**
ProximalPhalanxOutlineCorrect	0.84	0.80	0.81	0.79	0.81	0.82	**0.89**
PhalangesOutlinesCorrect	0.76	0.74	0.76	0.74	0.77	0.77	**0.79**
MiddlePhalanxOutlineCorrect	0.71	0.72	0.74	0.77	0.74	0.78	**0.81**
Herring	**0.63**	0.56	0.57	0.55	0.00	0.40	0.52
HandOutlines	**0.91**	0.86	0.87	0.90	0.80	0.80	0.83
GunPointOldVersusYoung	0.56	0.50	0.52	0.52	0.87	0.95	**0.97**
GunPointMaleVersusFemale	0.44	0.43	0.44	0.44	0.89	0.98	**0.99**
GunPointAgeSpan	0.44	0.44	0.44	0.44	0.86	0.88	**0.93**
GunPoint	0.69	0.68	0.67	0.68	0.77	0.93	**0.96**
DistalPhalanxOutlineCorrect	0.74	0.75	0.77	0.79	0.76	0.80	**0.81**
BirdChicken	0.56	0.54	0.60	0.63	0.51	**0.68**	**0.68**
BeetleFly	**0.83**	0.68	0.80	0.78	0.63	0.64	0.74

Table 1: F1-scores of the aggregated model. Results (mean) are obtained with five averaged runs.

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

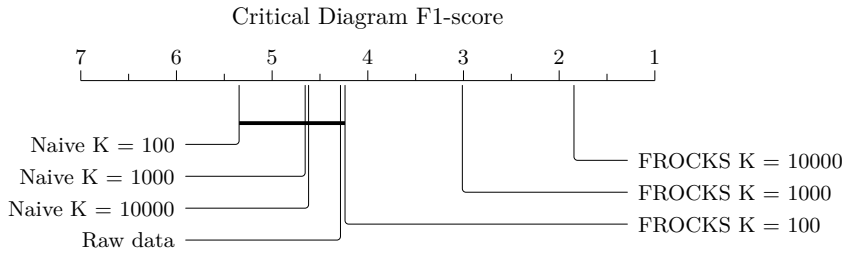

Fig. 1: Mean ranks for the naive approach and FROCKS for different K.

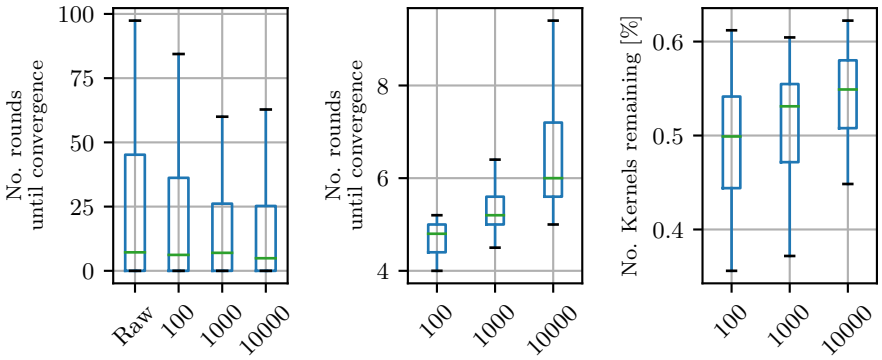

Fig. 2: Number of rounds until convergence (Fig. 2a and 2b), shown over all datasets, and the percentage of remaining kernels (2c). The x-axis indicates the number of kernels used to initialize.

Discussion: Table 1 shows the results of our experiments. FROCKS outperforms the naive approach and FedAvg on the raw data on most datasets. This suggests that the process of feature extraction benefits from exchanging the best-performing kernels and that averaging only the weights associated with kernels that are important for more than one client results in a more promising approach than naive parameter averaging. The critical difference diagram in Fig. 1 summarizes these results by showing each method's average rank (a lower rank more to the right is better). One can see that adopting a large number of kernels leads to better performance for both the naive federated approach and FROCKS, as expected. FROCKS offers the statistically best method for $K = 10\ 000$, followed by the second best choice of FROCKS with $K = 1\ 000$. Surprisingly, the naive approach is even worse than using raw data. We hypothesize that this is because, for each local client, a different set of feature transformations might be important, and the clients cannot converge to a common set of fea-

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

tures. From Fig. 2, we can see that FROCKS requires just a fraction of rounds to reach convergence compared to the naive approach. We hypothesize that this is because, in FROCKS, the federation can quickly reach a consensus on what feature transformations are relevant, whereas, in the naive approach, each client commits to all features at once. Last, it can be seen that FROCKS will roughly remove 50% of the initial features that are deemed unnecessary.

4 Conclusion and Future Work

This work presents FROCKS, a federated time series classification method based ROCKET features transformation. In our proposed approach, the federated aggregation is achieved by gathering the best-performing ROCKET kernels and averaging their corresponding weights if that kernel is top-performing for more than one client. We show the effectiveness of FROCKS through extensive experiments on the binary classification datasets of the UCR archive. Results indicate that our method outperforms the naive approach based on FedAvg and raw data. Furthermore, FROCKS requires just a fraction of rounds to converge compared to the naive approach. For future work, we aim to test FROCKS on multiclass datasets, to study its performance under some of the most challenging non-IID settings for FL, such as the label quantity skew, and to analyze the scalability of our proposed method by varying the number of parties of the federation.

References

[1] Anthony J. Bagnall et al. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Min. Knowl. Discov.*, 31(3):606–660, 2017.

[2] Alejandro Pasos Ruiz et al. The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Min. Knowl. Discov.*, 35(2):401–449, 2021.

[3] Angus Dempster, François Petitjean, and Geoffrey I. Webb. ROCKET: Exceptionally Fast and Accurate Time Series Classification using Random Convolutional Kernels. *Data Min. Knowl. Discov.*, 34(5):1454–1495, 2020.

[4] Brendan McMahan et al. Communication-efficient learning of deep networks from decentralized data. In *AISTATS 2017*. PMLR, 2017.

[5] Patrick Foley et al. Openfl: the open federated learning library. *Physics in Medicine & Biology*, 2022.

[6] Dau, Hoang Anh and others. The ucr time series classification archive, October 2018.