Neural Processing Letters (2021) 53:1995–2028
https://doi.org/10.1007/s11063-021-10484-z

Attention-Based Deep Gated Fully Convolutional End-to-End Architectures for Time Series Classification

Mehak Khan[1] · **Hongzhi Wang**[1] · **Alladoumbaye Ngueilbaye**[1]

Accepted: 27 February 2021 / Published online: 24 March 2021

Abstract
Time series classification (TSC) is one of the significant problems in the data mining community due to the wide class of domains involving the time series data. The TSC problem is being studied individually for univariate and multivariate using different datasets and methods. Subsequently, deep learning methods are more robust than other techniques and revolutionized many areas, including TSC. Therefore, in this study, we exploit the performance of attention mechanism, deep Gated Recurrent Unit (dGRU), *Squeeze-and-Excitation* (SE) block, and Fully Convolutional Network (FCN) in two end-to-end hybrid deep learning architectures, Att-dGRU-FCN and Att-dGRU-SE-FCN. The performance of the proposed models is evaluated in terms of classification testing error and f1-score. Extensive experiments and ablation study is carried out on multiple univariate and multivariate datasets from different domains to acquire the best performance of the proposed models. The proposed models show effective performance over other published methods, also do not require heavy data pre-processing, and small enough to be deployed on real-time systems.

Keywords Attention mechanism · Convolutional neural network · Squeeze-and-excitation · Gated recurrent unit · Univariate time series classification · Multivariate time series classification

1 Introduction

Statistical data analysis and machine learning are widely studied topics and hold a strong connection with various fields. The primary goal of Machine learning is to develop algorithms that can detect, sense, and learn phenomena as humans do or may be more efficient than humans in practical value. Although researchers have been working to accomplish this

✉ Mehak Khan
mehakkhan@hit.edu.cn

Hongzhi Wang
wangzh@hit.edu.cn

Alladoumbaye Ngueilbaye
angueilbaye@hit.edu.cn

[1] School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China

goal, they have used simple and complex algorithms. Less human interaction and effective performance are the main points in machine learning studies. A good representation of data could be needed to design a basic algorithm. Without the expert knowledge of data and the model, this simplified representation can be a challenging problem. Therefore, in such scenarios, it is better to design a model that can learn data by extracting attributes or features and transform the data according to the model's need by using different data-pre-processing techniques. First neural networks and then deep learning has been employed to solve this issue. The simplest deep learning architecture is perceptron, developed in the late 1950s. Later, multilayer architectures with limited learning abilities were proposed to fulfill the required necessities. For over 40 years, deep learning is evolving in different areas, including data mining, computer vision, speech recognition, and signal processing [1–3].

Time series data consists of a series of data points indexed over time order [4], and it is mainly found in two categories: univariate and multivariate. In univariate time series data, only one variable is varying over time, while in multivariate, multiple variables are varying over time. The time series data is ubiquitous, exists in many application domains such as statistics, pattern recognition, earthquake prediction, econometrics, astronomy, signal processing, control engineering, communication engineering, finance, weather forecasting, Internet of Things (IoT), and healthcare.

During the last two decades, substantial research has been conducted to solve the TSC problems. TSC is an open and challenging problem in the community of data mining, which is the task of classifying data points indexed over time and predict class labels. However, the published univariate and multivariate time series methods require heavy data pre-processing, feature crafting, refining, and fine-tuning to obtain better results.

This paper introduces two novel end-to-end deep learning architectures based on the attention mechanism, dGRU, SE block, and FCN, to tackle univariate and multivariate TSC problems. Besides, to validate the performance, we used a wide range of univariate and multivariate datasets, and the extensive study demonstrates the effectiveness of the proposed models. Moreover, to the best of our knowledge, this is the first work of exploiting attention mechanism, dGRU, FCN, and SE block together in hybrid models for TSC.

1.1 Contributions

The main contributions of this paper can be summarized as follows:

- Two efficient attention-based hybrid deep learning architectures are proposed for univariate and multivariate TSC problems.
- The attention mechanism, dGRU, SE block, and FCN, are exploited to propose these novel hybrid models. The attention mechanism is exploited to construct the depth of dGRU and model long term dependencies. In contrast, FCN is employed to extract the hierarchy of features. The SE block is useful in recalibrating feature maps as a whole and suppresses the less informative ones. Therefore, the SE block can help FCN to perform complex multivariate TSC tasks at a minimal additional computational cost and produce significant results.
- An ablation study is provided to demonstrate the insights and impact of each module of proposed models.
- The proposed models are validated on 85 univariate and 35 multivariate time series publically available datasets.

- As attested by comprehensive experimental results, the proposed models achieve superior performance over published methods across multiple datasets.

The remainder of this paper is organized as follows. In Sect. 2, we review the literature. In Sect. 3, we present our proposed models. Section 4 demonstrates the experimental settings, and in Sect. 5, we explain the results with discussion, and lastly, we conclude our work in Sect. 6.

2 Related Work

2.1 Univariate TSC

Several approaches have been proposed to solve the univariate TSC problem. These approaches can be categorized into four main categories; 1. Distance-based methods, 2. Feature-based methods, 3. Ensemble-based methods, and 4. Deep neural networks (DNNs).

Euclidean Distance (ED) and Dynamic Time Warping (DTW) [5] are the earliest distance-based baselines that work directly on the raw time-series data with some pre-defined similarity metrics to perform classification. Another most common approach is the combination of the DTW and k-Nearest Neighbor (k-NN) classifier. This approach is also known as the benchmark classifier for TSC.

Feature-based methods transform the set of features that represent the global or local time series patterns and are then handled by the classification algorithm. In feature-based methods, Bag-of-Words (Bow) [6], Bag-of-features framework (TSBF) [7], Bag-of-SFA-Symbols (BOSS) [8], BOSSVS [9], and Word extraction for time Series classification (WEASEL) [10] have shown benchmark performances.

Ensemble-based methods integrate different approaches to form a computationally efficient classifier. These methods also achieved state-of-the-art results on univariate TSC tasks. Some of the best methods are Proportional Elastic Ensemble (PROP) [11] that integrates 11 different elastic distance measures based methods using a weighted ensemble scheme, a flat collective of transform-based ensembles (COTE) [12] combines 35 classifiers using the features mined from frequency and time domains and transforms into a single classifier. Moreover, Hierarchical Vote Collective of Transformation-based Ensembles (HIVE-COTE) [13, 14] integrates different classifiers into the collective, including BOSS and Shapelet Transform, and builds a new classifier. Whereas, Time Series Combination of Heterogeneous and Integrated Embedding Forest (TS-CHIEF) [15] builds on proximity forest, dictionary-based and interval-based splitters. HIVE-COTE and TS-CHIEF are considered to be the most scalable ensemble-based classifiers for TSC.

Nowadays, deep learning methods have shown remarkable achievements in many fields, and these methods are also being exploited for various classification problems, including image classification, sleep stage classification, automatic modulation classification, and many more [16–22]. Accordingly, during the past few years, many researchers and practitioners have started deploying deep learning techniques for TSC problems. The DNNs have shown superior performance over other TSC approaches. Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and their variants have been widely used for various applications as well as for TSC problems. Among several deep learning proposed approaches, to name the latest benchmark classifiers, Wang et al. [23] presented three deep learning baselines, FCN), Residual Neural Network (ResNet), and Multilayer perceptron

(MLP). Later on, Karim et al. [24] proposed two hybrid deep neural networks, Long Short Term Memory Fully Convolutional Network (LSTM-FCN) and Attention LSTM-FCN (ALSTM-FCN). These models yield state-of-the-art results. In [25], the author proposed a hybrid model, GRU-FCN, motivated by [24], based on GRU instead of LSTM. Since GRU is more computationally efficient than LSTM, this approach has also shown competitive results over many previously published methods. Lately, some other introduced DNNs based classifiers are InceptionTime [26] and Random Convolutional Kernel Transform (ROCKET) [27]. InceptionTime is inspired by the Inception-v4 architecture and is based on deep CNNs. In comparison, ROCKET uses random convolutional kernels to transform time series and then use those transformed features to train a linear classifier. InceptionTime & ROCKET have shown significant performance for the TSC problem.

2.2 Multivariate TSC

Over the past years, multivariate TSC has attracted much attention, and a substantial amount of research has been carried to solve this problem. Multivariate time series data is produced more extensively than univariate data due to the broad range of applications dealing with the data that depends on multiple variables such as sensors, healthcare, and finance.

Among all the methods, the most common approach is to use DTW and k-NN together [28]. The word extraction for time series classification—multivariate unsupervised symbols and derivatives (WEASEL-MUSE) [29] and symbolic representation for multivariate time series (SMTS) [30] are other approaches that achieve state-of-the-art results on multivariate TSC problem.

Some promising deep learning algorithms for Multivariate TSC are Multi-Channel Deep Convolutional Neural Network (MC-DCNN) [31], Multivariate LSTM-FCN (MLSTM-FCN) [32], Multivariate ALSTM-FCN (MALSTM-FCN) [32] and Time Series Attentional Prototype Network (TapNet) [33]. MC-DCNN integrates the learned features from each channel and then feeds them into a MLP to perform classification. MLSTM-FCN & MALSTM-FCN claims current state-of-the-art results, also requires minimum pre-processing, and small enough to be deployed on memory-constrained systems. TapNet is introduced to handle the issue of limited labeled data in a multivariate TSC problem. They propose an attentional prototype network that trains latent features representations built on distances to class prototype with inadequate training labels.

3 Proposed Models

This section explains background components and the network architectures of two attention-based end-to-end models proposed in this study.

3.1 Att-dGRU-FCN

3.1.1 Attention Mechanism

The attention mechanism is one of the most dominant concepts in the deep learning community. It was primarily proposed for natural language processing [34] and then adopted by more domains such as computer vision [35], speech recognition, healthcare, and recommendation systems. It was an effort to implement the technique which can concentrate on selecting relevant and ignore irrelevant parts in a deep neural network, and this way, the model can pay attention to essential features. Mathematically, the attention mechanism can be described as follows.

$$c_t = \sum_{i=1}^{n} a_{t,i} h_i \tag{1}$$

$$a_{t,i} = \frac{exp(e_{t,i})}{\sum_{i=1}^{n} exp(e_{t,i})} \tag{2}$$

$$e_{t,i} = a(s_{t-1}, h_i) \tag{3}$$

where c_t is the context vector, depends on the sequence of annotations h_i, n denotes the input sequence length. The $a_{t,i}$ is a score assigned to the pair of input i and output t and $e_{t,i}$ is an attention weight.

3.1.2 Fully Convolutional Network (FCN)

For TSC, FCN was first used by Wang et al. [23] and proven as a robust deep learning baseline. In this study, FCN is exploited as a feature extractor, which is end-to-end and typically used for classification tasks. The FCN architecture consists of three *1D* kernels with sizes of 8, 5, and 3, respectively, without striding. All the convolutional layers are separately connected with batch normalization [36] and activation function. The final model is built by stacking three layers with the filter sizes of 128, 256, and 128, respectively. The output comes from the SoftMax layer.

Mathematically, the architecture can be explained as follows:

$$t = w \odot x + b \tag{4}$$

$$a = BN(t) \tag{5}$$

$$y = ReLU(a) \tag{6}$$

where $\odot$ shows the convolutional operator. *BN* denotes the batch normalization, and Rectified Linear Unit (*ReLU*) [37] is the activation function used in all the layers.

3.1.3 Gated Recurrent Unit (GRU)

A GRU is one of the RNN variants. It was first proposed by Cho et al. [38] to capture long-term dependencies on different time scales adaptively. It has a smaller architecture and requires fewer parameters compared to LSTM since it consists of only two gates: reset and update. Figure 1 shows the graphical illustration of the GRU.

The mathematical representation of the GRU can be directly translated into formulas from Eqs. 7–10.

$$z^{(t)} = \sigma(W_z x^{(t)} + U_z h^{(t-1)} + b_z) \tag{7}$$

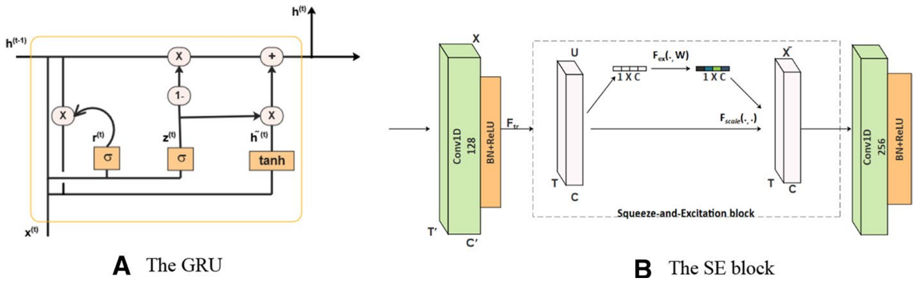


Fig. 1 **a** The GRU. **b** The SE block

$$r^{(t)} = \sigma(W_r x^{(t)} + U_r h^{(t-1)} + b_r) \tag{8}$$

$$\tilde{h}^{(t)} = tanh(W_x x^{(t)} + U_h(r^{(t)} e h^{(t-1)}) + b_h) \tag{9}$$

$$h^{(t)} = (1 - z^{(t)}) \odot h^{(t-1)} + z^{(t)} \odot \tilde{h}^{(t)} \tag{10}$$

where $x^{(t)}$ and $h^{(t)}$ are input and output vectors, $z^{(t)}$ and $r^{(t)}$ are the update and reset gates at a time-step t, respectively. W denotes feedforward weights, U represents recurrent weights, and b shows biases, respectively, while $\tilde{h}^{(t)}$ shows an output candidate activation function.

3.1.4 Integration of Attention Mechanism, dGRU, and FCN

The Att-dGRU-FCN is a two-stream hybrid model composed of three deep learning techniques; the Attention Mechanism, dGRU, and FCN. The first stream is the combination of attention mechanism and dGRU, while the second stream works with FCN, as shown in Fig. 2a. The attention mechanism is used to construct the depth of dGRU and model long term dependencies. By using attention with dGRU, the model can exploit the hierarchy of temporal features in the time series data. A GRU is simpler than LSTM for having fewer parameters, so it is computationally more efficient and needs a lesser amount of data to generalize. Moreover, the dGRU consists of two layers that can help the model learn more complex features from different datasets.

The Att-dGRU and FCN streams take the univariate input from two different perspectives:

The Att-dGRU takes the input and passes it through the masking layer to dGRU. Masking helps the Att-dGRU stream to handle different variable-length inputs. After the dGRU layers, we applied dropout (0.8) [39] for better generalization. Generalization helps the model prevent overfitting and improve the model's performance on different time series datasets. Right after the dropout, the attention mechanism is applied.

The FCN stream is used for feature extraction, which takes the input through the permute layer, commands the dimensions according to a given input, and passes it to the FCN block. The FCN block's architecture consists of three *1D* convolutional temporal blocks, with the kernel sizes 8, 5, and 3, respectively, with padding, while the filter number is 128, 256, 128, respectively. The He uniform variance scaling initializer [40] defines how to set the initial random weights for each convolutional block. BN and the ReLU activation function accompany each convolutional block. The BN gives uniformity to the training process's input and stability, which improves the generalization capability. ReLU is the most common activation

Fig. 2 **a** The network architecture of Att-dGRU-FCN. **b** The model graph of Att-dGRU-FCN using Keras library

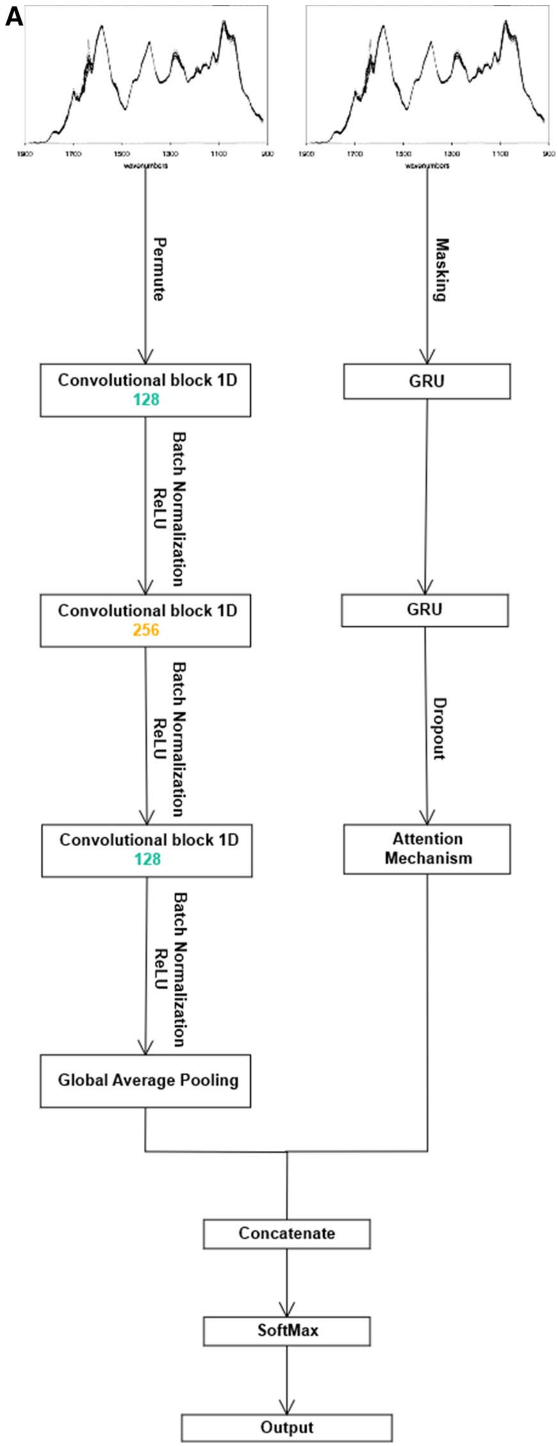
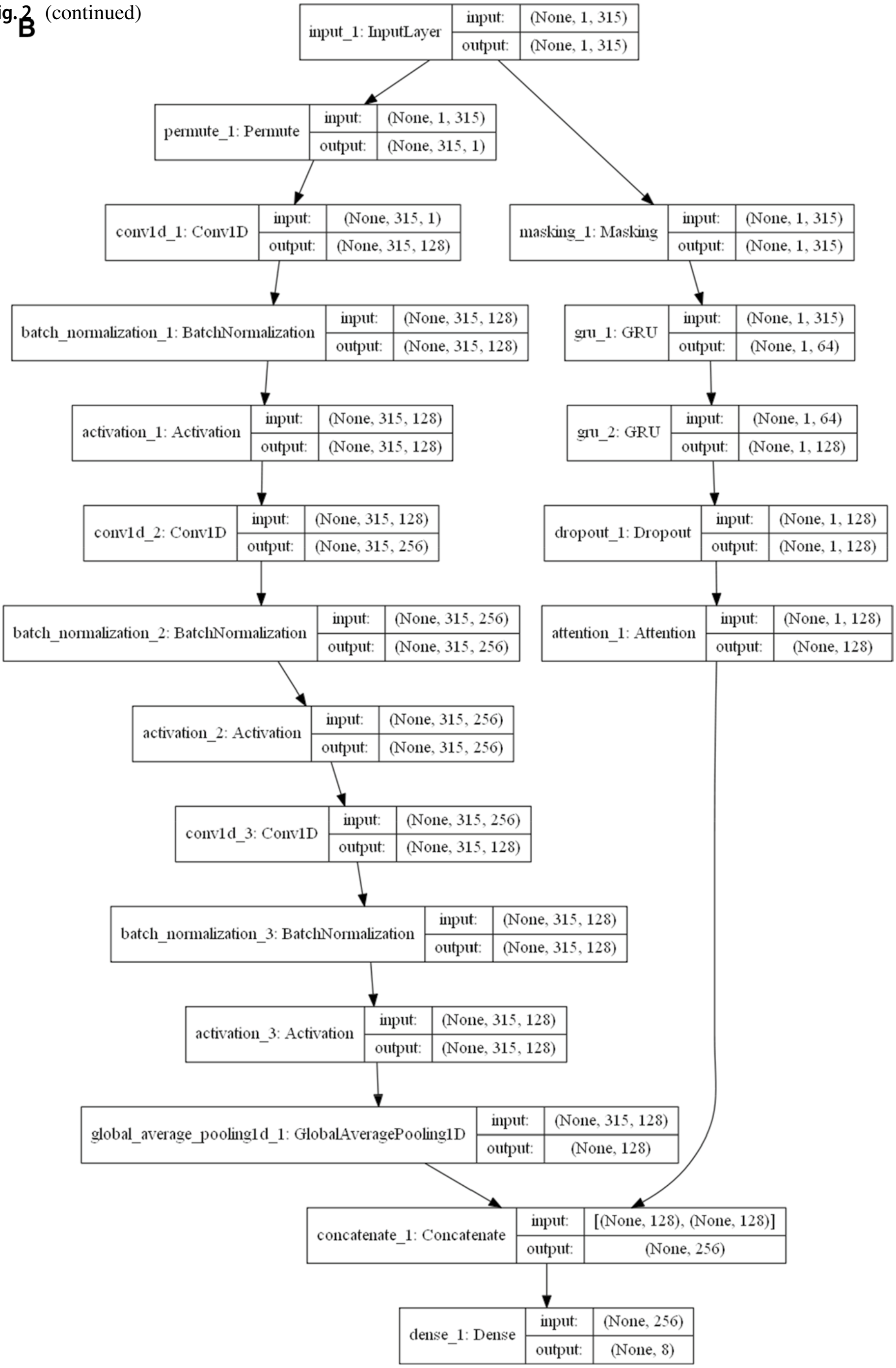


Fig. 2 B (continued)



function, especially for Convolution neural networks, which significantly improves performance. Later, the global average pooling [41] is applied to avoid overfitting by decreasing the model's overall number of trainable parameters. Each stream's output is concatenated and delivered to the SoftMax activation layer to receive the final output as predicted classes. The model graph built-in Keras library [42] is illustrated to portray a better understanding of our proposed univariate model, Fig. 2b.

The key idea to propose this model is: By using attention mechanism with dGRU, the model can exploit the hierarchy of temporal features in the time series data, while FCN is useful for feature extraction; therefore, infusion, this hybrid model can perform significantly better than many published methods.

3.2 Att-dGRU-SE-FCN

3.2.1 Squeeze-and-Excitation (SE) block

The SE block was first introduced by Hu et al. [43]. They used the SE block as a central building block for CNN. The SE block can significantly improve the CNN performance at a minimal additional computational cost. Therefore, in this study, we aim to leverage our proposed models' performance by integrating the SE block with FCN.

A SE block is a computational unit built upon F_{tr}. The F_{tr} is a transformation given to input $X \in \mathbb{R}^{W' \times H' \times C'}$ to generate a feature map $U \in \mathbb{R}^{W \times H \times C}$.

$$u_c = v_c * X = \sum_{s=1}^{C'} v_c^s * x^s. \tag{11}$$

Here $*$ denotes the convolution operator, $V_c = [v_c^1, v_c^2, \dots, v_c^{C'}]$, $X = [x^1, x^2, \dots, x^{C'}]$ and $U \in \mathbb{R}^{W \times H}$. A *2D* spatial kernel v_c^s is representing a single channel v_c that works as a corresponding channel of X. The basic working principles of the SE block can be explained in two steps: 1. Squeeze, and 2. Excitation. Figure 1b illustrates the basic structure of the SE block.

We compress the spatial dimensions into a channel specific descriptor by using global average pooling and generates channel-wise statistics in the squeeze operation. In time series data, the transformation output U can be shrunk through spatial dimension T for the computation of channel-wise statistics, $z \in \mathbb{R}^C$, and then the *c-th* element of z is calculated as follows:

$$z_c = F_{sq}(u_c) = \frac{1}{T} \sum_{t=1}^{T} u_c(t). \tag{12}$$

The *excitation* is the adaptive recalibration operation. In this operation, we make the full use of grouped information from squeeze operation by fully capturing channel-wise dependencies and attaining that the function must be flexible to learn the non-linear and non-mutually-exclusive association between several channels. It also employs the self-gating mechanism through a sigmoid activation function.

$$s = F_{ex}(z, W) = \sigma(g(z, W)) = \sigma(W_1 \delta(W_1, z)), \tag{13}$$

where δ denotes the ReLU activation functions: $W_1 \in \mathbb{R}^{\frac{C}{r} \times C}$ and $W_2 \in \mathbb{R}^{\frac{C}{r} \times C}$. W_1 and W_2 is used to optimize the model's complexity and help with the generalization. F_{ex} is a neural network, σ is the sigmoid function, and r denotes the reduction ratio.

Lastly, the output of the SE block is obtained after rescaling U with the activation s:

$$\tilde{x}_c = F_{scale}(u_c, s_c) = s_c \times u_c, \tag{14}$$

where $\tilde{X} = [\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_c]$ and $F_{scale}(u_c, s_c)$ refers to the channel-wise multiplication among the feature map $u_c \in \mathbb{R}^T$ and the scalar s_c.

3.2.2 Integration of Attention Mechanism, dGRU, SE block, and FCN

The network architecture of this proposed model is an extension of Att-dGRU-FCN. In this model, the FCN is not lone; it is integrated with the SE block in one of the streams, as illustrated in Fig. 3a. This is also a two-stream hybrid model, based on Att-dGRU and SE-FCN. The architecture of Att-dGRU and FCN remained identical, as explained in Sect. 3.1.4.

The SE-FCN stream takes the multivariate input through the permute layer with time steps Q and variables per time step M. The SE block follows the first two convolutional blocks. The SE block is exploited to strengthen the FCN performance by adaptively recalibrating the input feature maps. The additional parameters introduced due to the integration of the SE block into FCN can increase the model's size. The total number of parameters can be computed as:

$$\frac{2}{r} \sum_{s=1}^{S} N_s \cdot C_s^2, \tag{15}$$

where r is the reduction ratio, S denotes the number of stages, which is the collection of blocks operating on a standard spatial dimension, C_s refers to the dimension of the output channel in stage s, N_s depicts the number of repeated blocks for the stage s. Since FCN is kept constant, we can efficiently compute the additional parameters as $\frac{2}{16} * \left\{(128)^2 + (256)^2\right\} = 10240$ for the SE-FCN stream. Figure 3b demonstrates the model graph of this proposed multivariate model using the Keras library.

The key idea behind proposing this model is: the SE block is useful in recalibrating feature maps as a whole and suppresses the less informative ones; besides, FCN is already proven better for TSC as a baseline. Therefore, this model can show comparable performance for various Multivariate TSC tasks with an infusion with the Att-dGRU stream.

4 Experimental Settings

4.1 Datasets

4.1.1 Univariate TSC Datasets

To validate the performance on univariate datasets, we used the University of California Riverside (UCR) 2015 archive [44], which consists of 85 univariate datasets belong to different domains, including sensor, motion, image, stimulated, Electrocardiogram (ECG), spectro, and device. The sequence length among datasets varies from 24 to 2709 observations, and the number of classes differs from 2 to 60. The datasets are available pre-processed on the archive, so no further data pre-processing is required, Table 1.

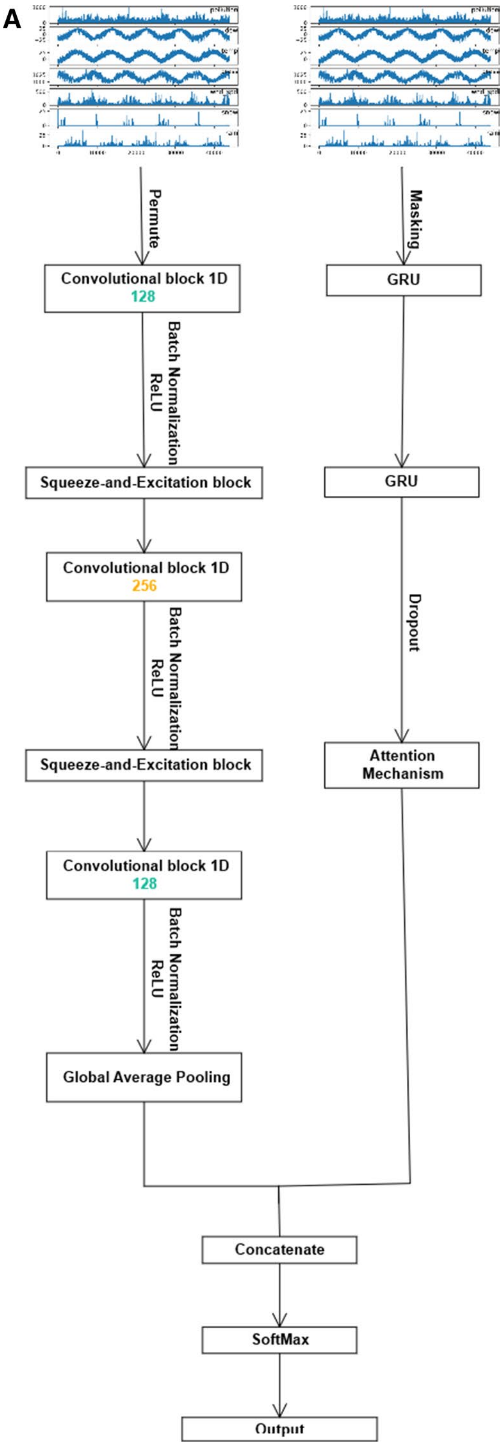
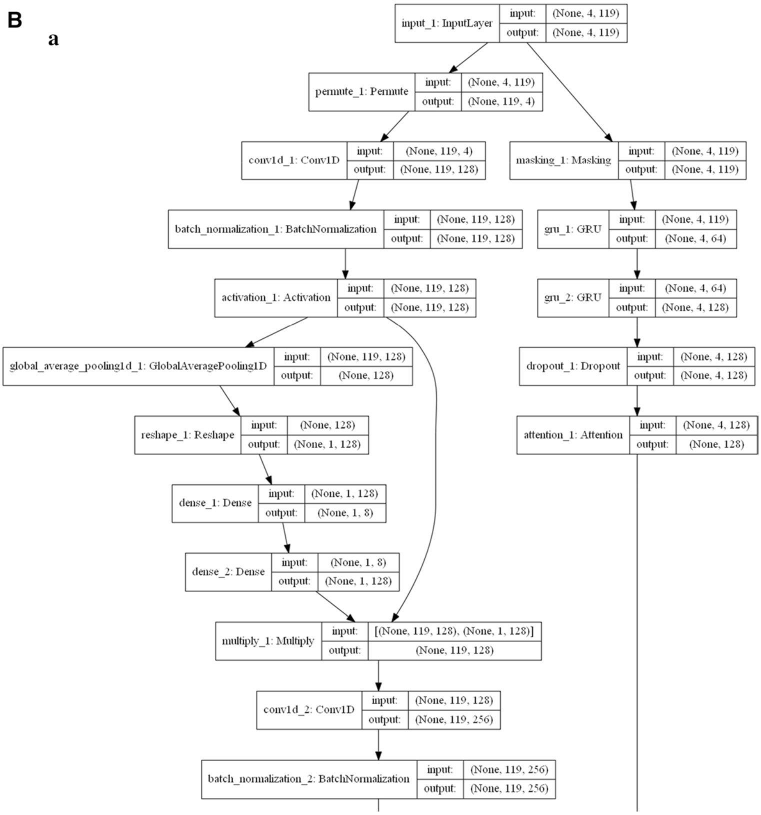


Fig. 3 **a** The network architecture of Att-dGRU-SE-FCN. **b** The model graph of Att-dGRU-SE-FCN using Keras library

Fig. 3 (continued)



4.1.2 Multivariate TSC Datasets

We tested our multivariate model on 35 multivariate datasets used and pre-processed by Karim et al. [32]. They collected these datasets from multiple sources [29, 45, 46] to perform different classification tasks, Table 2.

4.2 Training

During the training stage, these models use Adam Optimizer [47] with an initial learning rate of 0.001, which was later reduced to 0.0001. For the univariate model, the batch size is kept as 128 and trained with the epochs between 2000 and 3000. For the multivariate model, the batch size is also kept as 128 and epochs between 500 and 2000. The number of epochs fluctuates according to the size of the dataset. The dGRU cells are set as 64 and 128, respectively. The proposed models were tested 2–3 times to obtain the best accuracy. We used a single GPU GTX 1060, Keras library [42] with the TensorFlow [48] in the backend to train the proposed models.

Fig. 3 (continued)

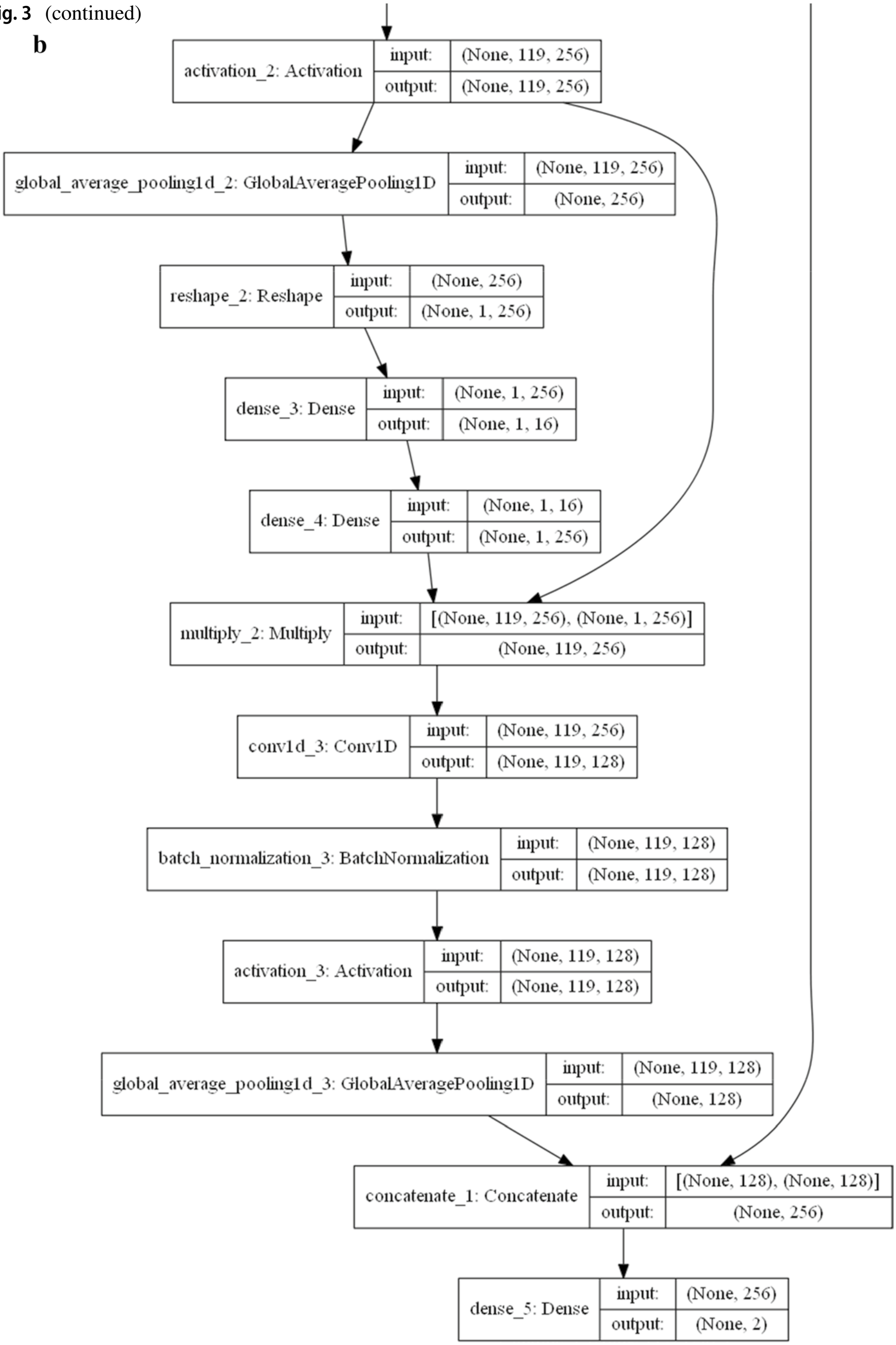


Table 1 The description of univariate TSC datasets [44]

Datasets	Type	Train	Test	Class	Length
Adiac	Image	390	391	37	176
ArrowHead	Image	36	175	3	251
Beef	Spectro	30	30	5	470
BeetleFly	Image	20	20	2	512
BirdChicken	Image	20	20	2	512
Car	Sensor	60	60	4	577
CBF	Simulated	30	900	3	128
ChlorineConcentration	Sensor	467	3840	3	166
CinCECGTorso	Sensor	40	1380	4	1639
Coffee	Spectro	28	28	2	286
Computers	Device	250	250	2	720
CricketX	Motion	390	390	12	300
CricketY	Motion	390	390	12	300
CricketZ	Motion	390	390	12	300
DiatomSizeReduction	Image	16	306	4	345
DistalPhalanxOutlineAgeGroup	Image	400	139	3	80
DistalPhalanxOutlineCorrect	Image	600	276	2	80
DistalPhalanxTW	Image	400	139	6	80
Earthquakes	Sensor	322	139	2	512
ECG200	ECG	100	100	2	96
ECG5000	ECG	500	4500	5	140
ECGFiveDays	ECG	23	861	2	136
ElectricDevices	Device	8926	7711	7	96
FaceAll	Image	560	1690	14	131
FaceFour	Image	24	88	4	350
FacesUCR	Image	200	2050	14	131
FiftyWords	Image	450	455	50	270
Fish	Image	175	175	7	463
FordA	Sensor	3601	1320	2	500
FordB	Sensor	3636	810	2	500
GunPoint	Motion	50	150	2	150
Ham	Spectro	109	105	2	431
HandOutlines	Image	1000	370	2	2709
Haptics	Motion	155	308	5	1092
Herring	Image	64	64	2	512
InlineSkate	Motion	100	550	7	1882
InsectWingbeatSound	Sensor	220	1980	11	256
ItalyPowerDemand	Sensor	67	1029	2	24
LargeKitchenAppliances	Device	375	375	3	720
Lightning2	Sensor	60	61	2	637
Lightning7	Sensor	70	73	7	319
Mallat	Simulated	55	2345	8	1024
Meat	Spectro	60	60	3	448
MedicalImages	Image	381	760	10	99

Table 1 (continued)

Datasets	Type	Train	Test	Class	Length
MiddlePhalanxOutlineAgeGroup	Image	400	154	3	80
MiddlePhalanxOutlineCorrect	Image	600	291	2	80
MiddlePhalanxTW	Image	399	154	6	80
MoteStrain	Sensor	20	1252	2	84
NonInvasiveFetalECGThorax1	ECG	1800	1965	42	750
NonInvasiveFetalECGThorax2	ECG	1800	1965	42	750
OliveOil	Spectro	30	30	4	570
OSULeaf	Image	200	242	6	427
PhalangesOutlinesCorrect	Image	1800	858	2	80
Phoneme	Sensor	214	1896	39	1024
Plane	Sensor	105	105	7	144
ProximalPhalanxOutlineAgeGroup	Image	400	205	3	80
ProximalPhalanxOutlineCorrect	Image	600	291	2	80
ProximalPhalanxTW	Image	400	205	6	80
RefrigerationDevices	Device	375	375	3	720
ScreenType	Device	375	375	3	720
ShapeletSim	Simulated	20	180	2	500
ShapesAll	Image	600	600	60	512
SmallKitchenAppliances	Device	375	375	3	720
SonyAIBORobotSurface1	Sensor	20	601	2	70
SonyAIBORobotSurface2	Sensor	27	953	2	65
StarLightCurves	Sensor	1000	8236	3	1024
Strawberry	Spectro	613	370	2	235
SwedishLeaf	Image	500	625	15	128
Symbols	Image	25	995	6	398
SyntheticControl	Simulated	300	300	6	60
ToeSegmentation1	Motion	40	228	2	277
ToeSegmentation2	Motion	36	130	2	343
Trace	Sensor	100	100	4	275
TwoLeadECG	ECG	23	1139	2	82
TwoPatterns	Simulated	1000	4000	4	128
UWaveGestureLibraryAll	Motion	896	3582	8	945
UWaveGestureLibraryX	Motion	896	3582	8	315
UWaveGestureLibraryY	Motion	896	3582	8	315
UWaveGestureLibraryZ	Motion	896	3582	8	315
Wafer	Sensor	1000	6164	2	152
Wine	Spectro	57	54	2	234
WordSynonyms	Image	267	638	25	270
Worms	Motion	181	77	5	900
WormsTwoClass	Motion	181	77	2	900
Yoga	Image	300	3000	2	426

Table 2 The description of multivariate TSC datasets [32]

Datasets	Number of classes	Number of variables	Maximum training length	Multivariate tasks	Train and test split
Arem	7	7	480	Activity recognition	50–50 split
Daily sport	19	45	125	Activity recognition	50–50 split
EEG	2	13	117	EEG classification	50–50 split
EEG2	2	64	256	EEG classification	20–80 split
Gesture phase	5	18	214	Gesture recognition	50–50 split
HAR	6	9	128	Activity recognition	71–29 split
HT sensor	3	11	5396	Food classification	50–50 split
Movement AAL	2	4	119	Movement classification	50–50 split
Occupancy	2	5	3758	Occupancy classification	35–65 split
Ozone	2	72	291	Weather classification	50–50 split
MSR activity	16	570	337	Activity recognition	5 ppl in train; rest in test
MSR action	20	570	100	Action recognition	5 ppl in train; rest in test
Cohn–Kanade AU-Coded expression CK	7	136	71	Facial expression classification	tenfold
Arabic voice	88	39	91	Speaker recognition	75–25 split
OHC	20	30	173	Handwriting classification	tenfold
ArabicDigits	10	13	93	Digit recognition	75–25 split
Auslan	95	22	96	Sign language recognition	44–56 split
Charactertrajectories	20	3	205	Handwriting classification	10–90 split
CMUsubject16	2	62	534	Action recognition	50–50 split
DigitShape	4	2	97	Action recognition	60–40 split
ECG	2	2	147	ECG classification	50–50 split
JapaneseVowels	9	12	26	Speech recognition	42–58 split
KickvsPunch	2	62	761	Action recognition	62–38 split
Libras	15	2	45	Sign language recognition	38–62 split
LP1	4	6	15	Robot failure recognition	43–57 split
LP2	5	6	15	Robot failure recognition	36–64 split
LP3	4	6	15	Robot failure recognition	36–64 split
LP4	3	6	15	Robot failure recognition	36–64 split
LP5	5	6	15	Robot failure recognition	39–61 split
NetFlow	2	4	994	Action recognition	60–40 split
PenDigits	10	2	8	Digit recognition	2–98 split

Table 2 (continued)

Datasets	Number of classes	Number of variables	Maximum training length	Multivariate tasks	Train and test split
Shapes	3	2	97	Action recognition	60–40 split
Uwave	8	3	315	Gesture recognition	20–80 split
Wafer	2	6	198	Manufacturing classification	25–75 split
WalkVsRun	2	62	1918	Action recognition	64–36 split

4.3 Evaluation Metrics

To evaluate the proposed models' performance, we used the classification testing error rate, f1-score, and the Mean Per Class Error (MPCE) method. A testing error rate results from training a classifier on new observations (test dataset), unseen by a model and not included in the training dataset. Besides, MPCE was introduced by Wang et al. [23] to evaluate the performance of a classifier on more than one dataset.

5 Results and Discussion

5.1 Univariate TSC

We compared Att-dGRU-FCN performance with the present state-of-the-art methods LSTM-FCN, ALSTM-FCN, along with FCN, MLP, RESNET as strong deep learning baselines. We also compared results with DTW and ED, which are traditional baselines for TSC. LSTM-FCN and ALSTM-FCN were trained from scratch to obtain their performance based on classification testing error rate and f1-score. Tables 3 and 4 demonstrate the results based on classification testing error and f1-score.

In terms of classification testing error, the proposed model, Att-dGRU-FCN, showed superior performance on 38 datasets with 0.029 MPCE score. LSTM-FCN and ALSTM-FCN depict the best performance on 21 and 28 datasets with the MPCE score of 0.032 and 0.033, respectively. The deep learning baselines FCN, ResNet, and MLP, win over 08, 14, and 03 datasets. The traditional baselines DTW and ED depict better performance over 06 and 02 datasets, Table 3. Figures 4 and 5 plots the MPCE and win rate on Att-dGRU-FCN, state-of-the-art, and baselines.

In contrast, Att-dGRU-FCN showed superior performance on 55 datasets, while the LSTM-FCN and ALSTM-FCN models show the best results over 23 and 24 datasets in terms of f1-score, Table 4.

The HandOutlines image type dataset is the largest among all the datasets in terms of sequence length (2709) and the smallest in terms of the number of classes (02). Over the HandOutlines dataset, Att-dGRU-FCN surpassed the performance over existing deep learning state-of-the-art methods and the baselines. Subsequently, over the ItalyPowerDemand, a sensor type of data, with the smallest size of sequence length (24) among all the datasets, Att-dGRU-FCN similarly showed superior performance over other methods.

Table 3 The proposed univariate model's classification testing error rate with present best deep learning methods and baselines on the 85 UCR archive datasets [44]

Datasets	Att-dGRU-FCN	LSTM-FCN	ALSTM-FCN	FCN	ResNet	MLP	DTW	ED
Adiac	**0.122**	0.140	0.125	0.143	0.174	0.248	0.396	0.389
ArrowHead	**0.102**	**0.102**	0.119	0.120	0.183	0.177	0.297	0.200
Beef	**0.066**	0.167	0.166	0.250	0.233	0.167	0.367	0.333
BeetleFly	**0.000**	0.050	**0.000**	0.050	0.200	0.150	0.300	0.250
BirdChicken	**0.000**	**0.000**	**0.000**	0.050	0.100	0.200	0.250	0.450
Car	**0.016**	0.033	0.050	0.083	0.067	0.167	0.267	0.267
CBF	0.002	0.002	0.004	**0.000**	0.006	0.140	0.003	0.148
ChlorineConcentration	0.166	0.191	0.176	0.157	0.172	**0.128**	0.352	0.350
CinCECGTorso	0.133	0.155	0.115	0.187	0.229	0.158	0.349	**0.103**
Coffee	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
Computers	0.136	0.136	**0.123**	0.152	0.176	0.460	0.300	0.424
CricketX	0.248	0.193	0.202	0.185	**0.179**	0.431	0.246	0.423
CricketY	0.246	**0.183**	0.185	0.208	0.195	0.405	0.256	0.433
CricketZ	0.235	0.184	**0.175**	0.187	0.187	0.408	0.246	0.413
DiatomSizeReduction	0.039	0.046	0.063	0.070	0.069	0.036	**0.033**	0.065
DistalPhalanxOutlineAgeGroup	**0.107**	0.132	0.137	0.165	0.202	0.173	0.230	0.374
DistalPhalanxOutlineCorrect	**0.079**	0.166	0.163	0.188	0.180	0.190	0.283	0.283
DistalPhalanxTW	**0.180**	0.185	0.185	0.210	0.260	0.253	0.410	0.367
Earthquakes	0.173	0.177	**0.170**	0.199	0.214	0.208	0.281	0.288
ECG200	**0.079**	0.080	0.090	0.100	0.130	0.080	0.230	0.120
ECG5000	0.054	0.053	**0.052**	0.059	0.069	0.065	0.076	0.075
ECGFiveDays	**0.002**	0.011	0.009	0.015	0.045	0.030	0.232	0.203
ElectricDevices	**0.037**	**0.037**	**0.037**	0.277	0.272	0.420	0.399	0.449
FaceAll	0.047	0.060	**0.045**	0.071	0.166	0.115	0.192	0.286
FaceFour	**0.056**	**0.057**	**0.057**	0.068	0.068	0.170	0.171	0.216
FacesUCR	0.077	0.071	0.057	0.052	**0.042**	0.185	0.095	0.231
FiftyWords	0.281	0.196	**0.176**	0.321	0.273	0.288	0.301	0.369

Table 3 (continued)

Datasets	Att-dGRU-FCN	LSTM-FCN	ALSTM-FCN	FCN	ResNet	MLP	DTW	ED
Fish	0.022	0.017	0.023	0.029	**0.011**	0.126	0.177	0.217
FordA	0.115	**0.072**	0.073	0.094	**0.072**	0.231	0.444	0.335
FordB	0.106	0.088	**0.081**	0.117	0.100	0.371	0.380	0.394
GunPoint	**0.000**	**0.000**	**0.000**	**0.000**	0.007	0.067	0.093	0.087
Ham	**0.190**	0.209	0.228	0.238	0.219	0.286	0.533	0.400
HandOutlines	**0.098**	0.113	0.358	0.224	0.139	0.193	0.119	0.138
Haptics	0.464	**0.425**	0.435	0.449	0.494	0.539	0.623	0.630
Herring	**0.218**	0.250	0.265	0.297	0.406	0.313	0.469	0.484
InlineSkate	0.538	0.534	**0.507**	0.589	0.635	0.649	0.616	0.658
InsectWingbeatSound	0.352	0.342	**0.329**	0.598	0.469	0.369	0.643	0.438
ItalyPowerDemand	**0.028**	0.037	0.032	0.030	0.040	0.034	0.050	0.045
LargeKitchenAppliances	0.133	0.090	**0.083**	0.104	0.107	0.520	0.205	0.507
Lightning2	0.180	0.197	0.213	0.197	0.246	0.279	**0.131**	0.246
Lightning7	0.178	0.164	0.178	**0.137**	0.164	0.356	0.274	0.427
Mallat	0.020	0.019	**0.016**	0.020	0.021	0.064	0.066	0.086
Meat	0.033	0.116	0.033	0.033	**0.000**	0.067	0.067	0.067
MedicalImages	0.201	**0.199**	0.204	0.208	0.228	0.271	0.263	0.316
MiddlePhalanxOutlineAgeGroup	**0.182**	0.188	0.189	0.232	0.240	0.265	0.500	0.481
MiddlePhalanxOutlineCorrect	**0.160**	**0.160**	0.163	0.205	0.207	0.240	0.302	0.234
MiddlePhalanxTW	**0.348**	0.383	0.373	0.388	0.393	0.391	0.494	0.487
MoteStrain	0.056	0.061	0.064	**0.050**	0.105	0.131	0.165	0.121
NonInvasiveFetalECGThorax1	0.035	0.035	**0.025**	0.039	0.052	0.058	0.210	0.171
NonInvasiveFetalECGThorax2	0.037	0.038	**0.034**	0.045	0.049	0.057	0.135	0.120
OliveOil	**0.066**	0.133	0.067	0.167	0.133	0.600	0.167	0.133
OSULeaf	0.041	**0.004**	**0.004**	0.012	0.021	0.430	0.409	0.479

Table 3 (continued)

Datasets	Att-dGRU-FCN	LSTM-FCN	ALSTM-FCN	FCN	ResNet	MLP	DTW	ED
PhalangesOutlinesCorrect	**0.168**	0.177	0.170	0.174	0.175	0.170	0.272	0.239
Phoneme	0.725	0.650	**0.640**	0.655	0.676	0.902	0.772	0.891
Plane	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	0.019	**0.000**	0.038
ProximalPhalanxOutlineAgeGroup	**0.102**	0.117	0.107	0.151	0.151	0.176	0.195	0.215
ProximalPhalanxOutlineCorrect	0.072	**0.065**	0.075	0.100	0.082	0.113	0.217	0.192
ProximalPhalanxTW	**0.160**	0.167	0.173	0.190	0.193	0.203	0.244	0.293
RefrigerationDevices	0.426	**0.421**	0.429	0.467	0.472	0.629	0.536	0.605
ScreenType	0.343	0.341	0.328	0.333	**0.293**	0.592	0.603	0.640
ShapeletSim	**0.000**	0.011	0.011	0.133	**0.000**	0.517	0.350	0.461
ShapesAll	0.103	0.098	0.100	0.102	**0.088**	0.225	0.232	0.248
SmallKitchenAppliances	**0.181**	0.184	0.203	0.197	0.203	0.611	0.357	0.659
SonyAIBORobotSurface1	**0.009**	0.018	0.030	0.032	0.015	0.273	0.275	0.305
SonyAIBORobotSurface2	**0.017**	0.022	0.025	0.038	0.038	0.161	0.169	0.141
StarLightCurves	0.025	0.024	**0.023**	0.033	0.025	0.043	0.093	0.151
Strawberry	**0.000**	0.013	0.013	0.031	0.042	0.033	0.059	0.054
SwedishLeaf	0.023	0.021	**0.014**	0.034	0.042	0.107	0.208	0.211
Symbols	0.026	0.016	**0.013**	0.038	0.128	0.147	0.050	0.101
SyntheticControl	0.006	0.003	0.006	0.010	**0.000**	0.050	0.007	0.120
ToeSegmentation1	0.026	**0.013**	**0.013**	0.031	0.035	0.399	0.228	0.320
ToeSegmentation2	**0.069**	0.084	0.077	0.085	0.138	0.254	0.162	0.192
Trace	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	0.180	**0.000**	0.240
TwoLeadECG	**0.000**	0.001	0.001	**0.000**	**0.000**	0.147	0.096	0.253
TwoPatterns	0.005	0.003	0.003	0.103	**0.000**	0.114	**0.000**	0.093
UWaveGestureLibraryAll	0.050	0.096	0.107	0.174	0.132	**0.046**	0.108	0.052
UWaveGestureLibraryX	0.187	**0.151**	0.152	0.246	0.213	0.232	0.273	0.261

Table 3 (continued)

Datasets	Att-dGRU-FCN	LSTM-FCN	ALSTM-FCN	FCN	ResNet	MLP	DTW	ED
UWaveGestureLibraryY	0.282	**0.233**	0.234	0.275	0.332	0.297	0.366	0.338
UWaveGestureLibraryZ	0.246	0.203	**0.202**	0.271	0.245	0.295	0.342	0.350
Wafer	**0.001**	**0.001**	0.002	0.003	0.003	0.004	0.020	0.005
Wine	**0.055**	0.111	0.111	0.111	0.204	0.204	0.426	0.389
WordSynonyms	0.391	**0.329**	0.332	0.420	0.368	0.406	0.351	0.382
Worms	0.342	**0.298**	0.320	0.331	0.381	0.657	0.416	0.545
WormsTwoClass	**0.193**	0.215	0.198	0.271	0.265	0.403	0.377	0.390
Yoga	0.099	0.082	**0.081**	0.155	0.142	0.145	0.164	0.170
Win	**38**	21	28	08	14	03	06	02
MPCE	**0.029**	0.032	0.033	0.039	0.042	0.068	0.073	0.081

The instances in bold text depicts best performances

Table 4 The f1-score of the proposed univariate model with present best-known deep learning methods on the 85 UCR archive datasets

Datasets	Att-dGRU-FCN	LSTM-FCN	ALSTM-FCN
Adiac	**0.825**	0.770	0.780
ArrowHead	**0.712**	0.694	0.695
Beef	**0.935**	0.873	0.765
BeetleFly	**1.000**	**1.000**	0.949
BirdChicken	**1.000**	**1.000**	**1.000**
Car	**0.984**	0.952	0.947
CBF	**0.997**	0.994	0.989
ChlorineConcentration	0.774	**0.791**	0.767
CinCECGTorso	**0.863**	0.321	0.375
Coffee	**1.000**	**1.000**	**1.000**
Computers	0.488	**0.914**	0.913
CricketX	0.744	0.782	**0.784**
CricketY	0.737	**0.786**	0.776
CricketZ	0.737	**0.778**	0.761
DiatomSizeReduction	**0.942**	0.926	0.935
DistalPhalanxOutlineAgeGroup	0.625	0.614	**0.636**
DistalPhalanxOutlineCorrect	**0.901**	0.804	0.813
DistalPhalanxTW	**0.503**	0.469	0.479
Earthquakes	**0.516**	0.466	0.466
ECG200	**0.914**	0.900	0.909
ECG5000	**0.267**	0.251	0.263
ECGFiveDays	**0.998**	0.991	0.991
ElectricDevices	0.194	0.196	**0.197**
FaceAll	**0.137**	0.134	0.136
FaceFour	0.922	**0.949**	**0.949**
FacesUCR	0.872	**0.898**	0.896
FiftyWords	**0.393**	0.330	0.353
Fish	0.961	**0.964**	0.957
FordA	0.883	**0.928**	**0.928**
FordB	0.893	**0.930**	0.929
GunPoint	**1.000**	**1.000**	**1.000**
Ham	**0.810**	0.788	0.770
HandOutlines	**0.889**	0.873	0.866
Haptics	0.498	**0.523**	0.515
Herring	**0.752**	0.722	0.694
InlineSkate	0.438	**0.474**	0.446
InsectWingbeatSound	**0.632**	0.432	0.410
ItalyPowerDemand	**0.972**	0.970	**0.972**
LargeKitchenAppliances	**0.410**	0.407	**0.410**
Lightning2	**0.819**	0.767	0.767
Lightning7	0.772	0.833	**0.858**
Mallat	**0.976**	0.970	0.971
Meat	0.957	0.870	**0.973**
MedicalImages	0.700	0.686	**0.701**

Table 4 (continued)

Datasets	Att-dGRU-FCN	LSTM-FCN	ALSTM-FCN
MiddlePhalanxOutlineAgeGroup	**0.891**	0.347	0.445
MiddlePhalanxOutlineCorrect	0.820	**0.821**	0.819
MiddlePhalanxTW	0.227	0.314	**0.320**
MoteStrain	**0.942**	0.920	0.915
NonInvasiveFetalECGThorax1	**0.910**	0.908	0.905
NonInvasiveFetalECGThorax2	**0.903**	0.896	0.894
OliveOil	0.812	0.611	**0.885**
OSULeaf	0.957	0.979	**0.988**
PhalangesOutlinesCorrect	**0.810**	0.803	0.809
Phoneme	0.022	**0.026**	**0.026**
Plane	**0.995**	0.888	0.882
ProximalPhalanxOutlineAgeGroup	**0.611**	0.594	0.436
ProximalPhalanxOutlineCorrect	0.898	**0.904**	0.896
ProximalPhalanxTW	**0.524**	0.504	0.469
RefrigerationDevices	0.214	**0.241**	**0.241**
ScreenType	**0.314**	0.302	0.308
ShapeletSim	**0.856**	0.842	0.842
ShapesAll	**0.202**	0.108	0.107
SmallKitchenAppliances	0.361	0.361	**0.370**
SonyAIBORobotSurface1	**0.990**	0.974	0.983
SonyAIBORobotSurface2	**0.981**	0.978	0.977
StarLightCurves	**0.963**	0.961	0.962
Strawberry	**0.865**	0.818	0.818
SwedishLeaf	0.806	0.801	**0.811**
Symbols	0.970	0.982	**0.974**
SyntheticControl	**0.521**	0.516	0.511
ToeSegmentation1	0.729	**0.746**	**0.746**
ToeSegmentation2	**0.579**	0.563	0.577
Trace	**0.986**	**0.986**	0.983
TwoLeadECG	**1.000**	**0.999**	0.999
TwoPatterns	**0.995**	0.989	0.971
UWaveGestureLibraryAll	**0.948**	0.766	0.754
UWaveGestureLibraryX	**0.796**	0.654	0.659
UWaveGestureLibraryY	**0.704**	0.695	0.686
UWaveGestureLibraryZ	**0.744**	0.739	0.743
Wafer	0.994	**0.996**	**0.996**
Wine	**0.944**	0.887	0.887
WordSynonyms	**0.375**	0.327	0.345
Worms	**0.578**	0.423	0.425
WormsTwoClass	**0.791**	0.525	0.542
Yoga	0.898	0.906	**0.914**
Win	**55**	23	24

The instances in bold text depicts best performances

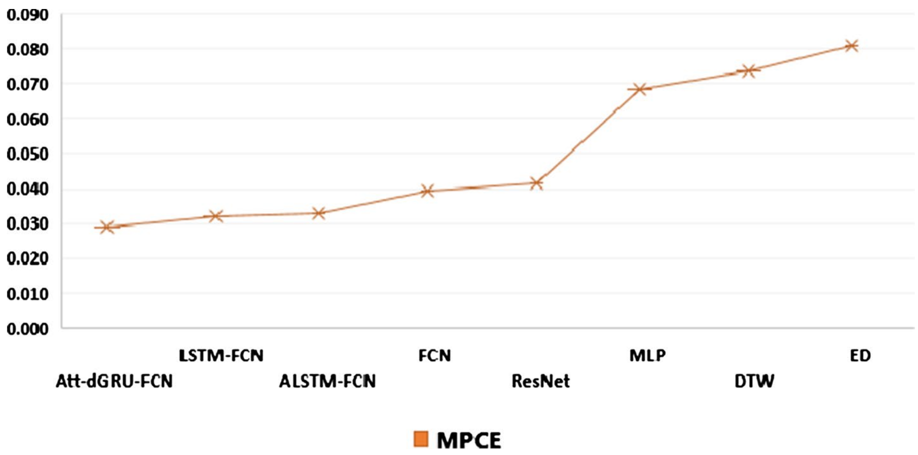


Fig. 4 The MPCE score on proposed univariate model Att-dGRU-FCN, state-of-the-art and baselines

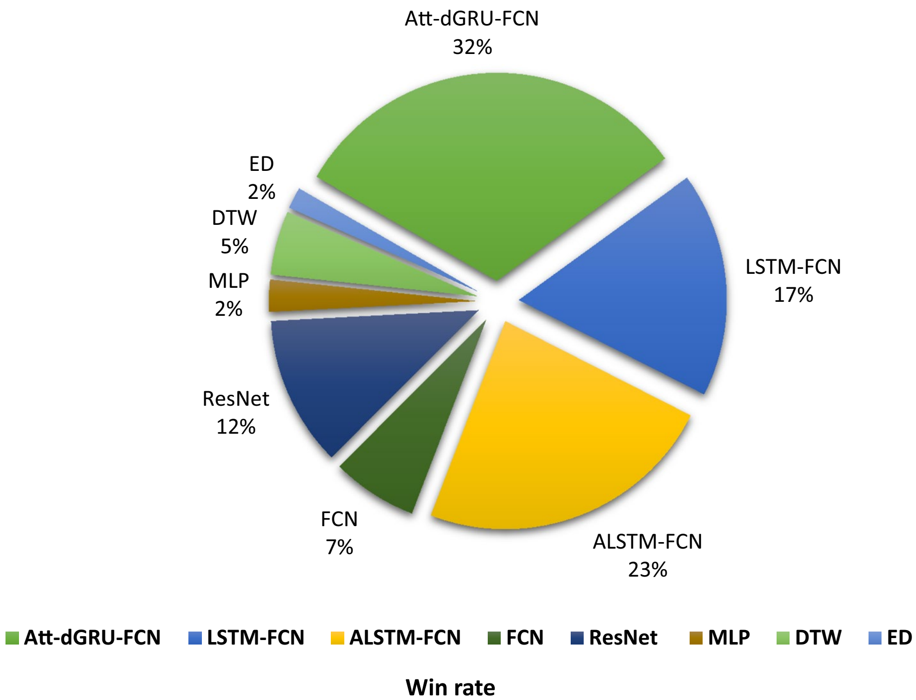


Fig. 5 The win rate on the proposed univariate model Att-dGRU-FCN, state-of-the-art and baselines

Figure 6 shows the training and validation testing error loss over Car (Sensor), ItalyPowerDemand (Sensor), Herring (Image), and MiddlePhalanxTW (Image), Coffee (Spectro), and SmallKitchenAppliances (Device) datasets. These figures illustrate how a classifier's performance on test data is not the same as training data.

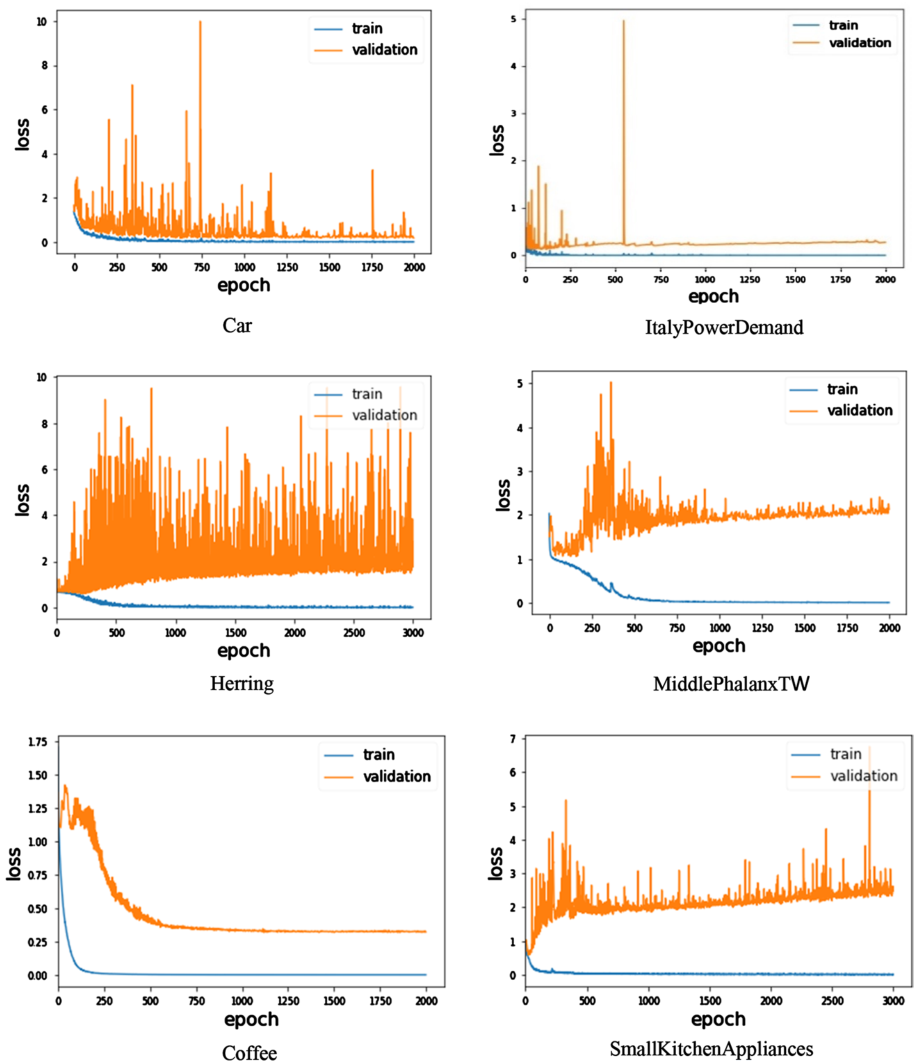


Fig. 6 The training and validation testing error loss of univariate model Att-dGRU-FCN on multiple datasets

Based on comprehensive experimental results, it is evident that the proposed model, Att-dGRU-FCN performance is significantly better than other state-of-the-art methods and the baselines.

5.2 Multivariate TSC

For performance comparison, we compared the proposed multivariate model, Att-dGRU-SE-FCN, with current state-of-the-art methods, MLSTM-FCN and MALSTM-FCN [32], and the deep learning baseline, FCN. MLSTM-FCN, MALSTM-FCN, and

Table 5 The proposed multivariate model's classification testing error rate with present best-known deep learning methods on 35 datasets

Dataset	Att-dGRU-SE-FCN	MLSTM-FCN	MALSTM-FCN	FCN
Arem	**0.128**	**0.128**	**0.128**	**0.128**
Daily Sport	0.005	**0.004**	0.006	0.005
EEG	0.437	0.437	**0.421**	0.500
EEG2	0.090	0.088	0.088	**0.062**
Gesture Phase	0.484	0.515	0.484	**0.474**
HAR	0.042	0.040	**0.035**	0.056
HT Sensor	**0.140**	0.399	0.280	0.380
Movement AAL	**0.229**	**0.229**	**0.229**	**0.229**
Occupancy	0.328	**0.197**	0.395	0.395
Ozone	**0.202**	0.243	0.208	0.266
MSR Activity	**0.425**	0.431	0.494	0.438
MSR Action	0.239	0.279	0.319	**0.235**
CK+	**0.071**	**0.071**	**0.071**	0.107
Arabic-Voice	0.024	0.024	0.020	**0.018**
OHC	**0.003**	0.007	0.007	**0.003**
ArabicDigits	**0.004**	0.007	0.009	0.007
Auslan	0.041	0.061	0.042	**0.035**
CharacterTrajectories	**0.006**	**0.006**	0.008	0.017
CMUsubject16	**0.000**	**0.000**	**0.000**	**0.000**
DigitShapes	**0.000**	**0.000**	**0.000**	**0.000**
ECG	**0.110**	0.149	0.139	0.149
JapaneseVowels	**0.005**	**0.005**	0.008	**0.005**
KickvsPunch	**0.000**	0.100	0.100	0.100
Libras	**0.020**	0.023	0.034	0.034
LP1	**0.140**	0.160	0.160	0.180
LP2	0.199	**0.166**	0.233	0.199
LP3	0.266	0.266	**0.233**	0.366
LP4	**0.079**	0.120	**0.079**	0.133
LP5	**0.329**	0.360	0.329	0.399
NetFlow	0.073	0.058	0.088	**0.045**
PenDigits	**0.035**	0.036	**0.035**	**0.035**
Shapes	**0.000**	**0.000**	**0.000**	**0.000**
UWave	**0.020**	0.024	0.023	0.023
Wafer	**0.008**	**0.008**	0.011	0.011
WalkvsRun	**0.000**	**0.000**	**0.000**	**0.000**
Win	**23**	13	12	15
MPCE	**0.034**	0.038	0.039	0.043

The instances in bold text depicts best performances

FCN were trained from scratch to obtain their performance based on classification

Table 6 The f1-score of the proposed multivariate model with present best-known deep learning methods on 35 datasets

Datasets	Att-dGRU-SE-FCN	MLSTM-FCN	MALSTM-FCN	FCN
Arem	**0.872**	**0.872**	**0.872**	**0.872**
Daily Sport	**0.995**	0.995	0.994	**0.995**
EEG	0.563	0.563	**0.578**	0.500
EEG2	0.910	0.912	0.912	**0.938**
Gesture Phase	0.509	0.485	0.520	**0.532**
HAR	0.957	0.960	**0.965**	0.944
HT Sensor	**0.860**	0.600	0.694	0.626
Movement AAL	**0.771**	**0.771**	**0.771**	**0.771**
Occupancy	0.671	**0.803**	0.605	0.605
Ozone	**0.798**	0.757	0.792	0.734
MSR Activity	0.573	0.571	0.518	**0.581**
MSR Action	**0.767**	0.724	0.689	0.748
CK+	**0.929**	**0.929**	**0.929**	0.893
Arabic-Voice	0.976	0.976	0.979	**0.982**
OHC	**0.996**	0.993	0.993	**0.996**
ArabicDigits	**0.996**	0.993	0.990	0.992
Auslan	0.961	0.939	0.957	**0.964**
CharacterTrajectories	**0.993**	0.993	0.991	0.985
CMUsubject16	**1.000**	**1.000**	**1.000**	**1.000**
DigitShapes	**1.000**	**1.000**	**1.000**	**1.000**
ECG	**0.890**	0.850	0.860	0.850
JapaneseVowels	**0.995**	0.993	0.992	0.993
KickvsPunch	**1.000**	0.900	0.900	0.900
Libras	**0.979**	0.971	0.966	0.967
LP1	**0.860**	0.840	0.840	0.820
LP2	0.800	**0.814**	0.780	0.780
LP3	0.746	0.724	**0.767**	0.633
LP4	**0.920**	0.872	**0.920**	0.872
LP5	**0.677**	0.643	0.660	0.600
NetFlow	0.927	0.942	0.912	**0.955**
PenDigits	0.965	0.965	0.965	**0.966**
Shapes	**1.000**	**1.000**	**1.000**	**1.000**
UWave	**0.980**	0.977	0.977	0.977
Wafer	**0.992**	**0.992**	0.989	0.989
WalkvsRun	**1.000**	**1.000**	**1.000**	**1.000**
Wins	**23**	10	11	15

The instances in bold text depicts best performances

testing error rate and f1-score. Tables 5 and 6 illustrate the results based on classification testing error and f1-score.

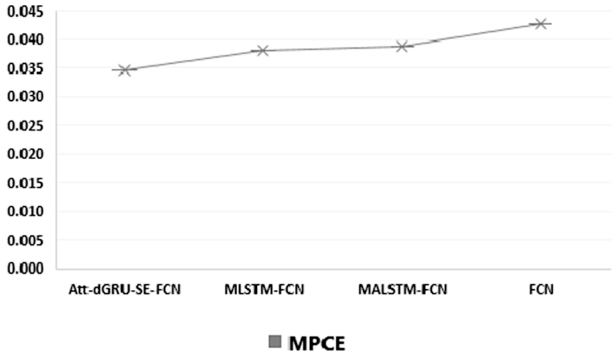


Fig. 7 The MPCE score on the proposed multivariate model Att-dGRU-SE-FCN, state-of-the-art and a baseline

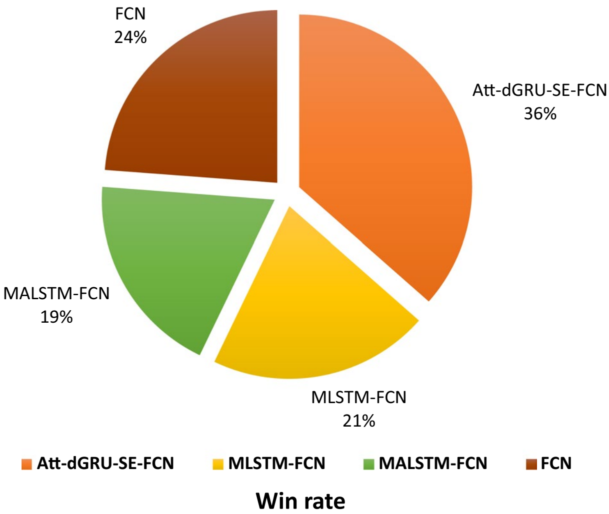


Fig. 8 The win rate on the proposed multivariate model Att-dGRU-SE-FCN, state-of-the-art and a baseline

Regarding classification testing error, Att-dGRU-SE-FCN showed the best performance over 23 out of 35 datasets with the MPCE score of 0.034. At the same time, MLSTM-FCN and MALSTM-FCN depict better performance on 13 and 12 datasets, with 0.038 and 0.039 MPCE score, respectively. The deep learning baseline FCN wins over 15 datasets with 0.043 MPCE score, Table 5. Figures 7 and 8 plots the MPCE and win rate on Att-dGRU-SE-FCN, state-of-the-art, and baseline.

Att-dGRU-SE-FCN also wins over 23 datasets, while the other methods, MLSTM-FCN and MALSTM-FCN, showed better performance on 10 and 11 datasets in terms of f1-score. The FCN wins over 15 datasets, Table 6.

All the methods depict similar performance on AREM (Activity recognition) dataset with 0.128 testing error rate and 0.872 f1-score. FCN has shown superior performance than other present best methods as a deep learning baseline. Figure 9 represents the training and validation testing error over the CMUsubject16, KickvsPunch (Action Recognition), LP1 (Robot Failure Recognition), Ozone (Weather Classification), JapaneseVowels (Speech Recognition), OHC (Handwriting classification), ArabicDigits (Digit recognition), and Libras (Sign Language Recognition) datasets. Figures 6 and 9 depicts that the dropout mechanism works relatively better on multivariate than univariate datasets.

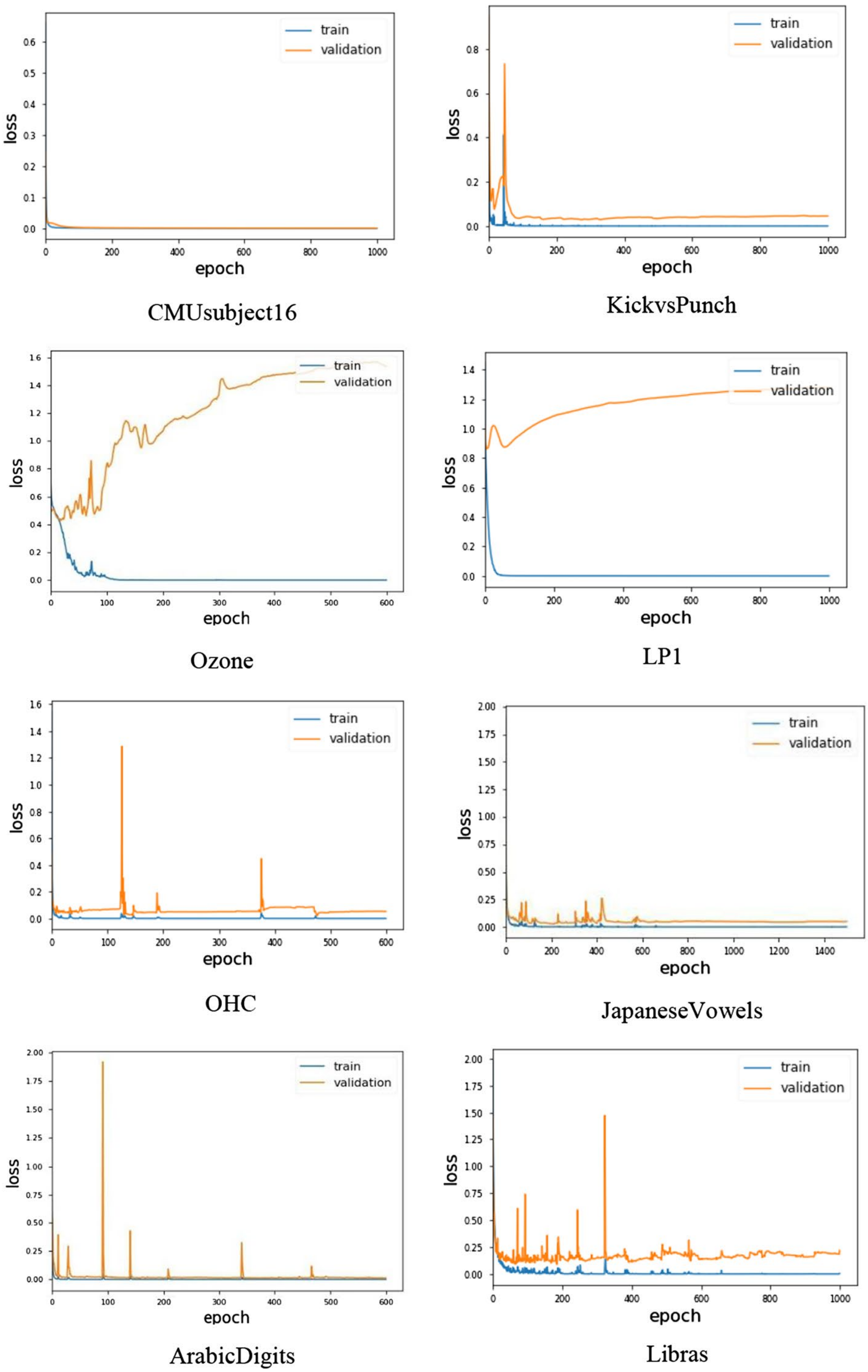


Fig. 9 The training and validation testing error loss of multivariate model Att-dGRU-SE-FCN on multiple datasets

Table 7 The ablation study on Att-dGRU-FCN model

Datasets	GRU	Att-GRU	dGRU	Att-dGRU	FCN	dGRU-FCN	Att-dGRU-FCN
Testing error rate							
ArrowHead	0.228	*0.280*	0.245	0.246	0.120	**0.102**	**0.102**
Beef	0.200	*0.333*	0.133	0.300	0.250	0.233	**0.066**
BeetleFly	*0.200*	0.100	0.050	0.100	0.050	0.050	**0.000**
Car	0.183	0.300	0.183	*0.417*	0.083	0.050	**0.016**
CBF	*0.140*	0.133	0.090	0.127	**0.000**	0.009	0.002
F1-score							
Adiac	*0.000*	*0.000*	0.016	0.021	0.740	0.761	**0.825**
ArrowHead	0.561	*0.447*	0.542	0.534	0.653	0.689	0.712
Beef	0.342	*0.182*	0.737	0.312	0.615	0.691	**0.935**
BeetleFly	*0.798*	0.899	0.950	0.899	0.950	0.950	**1.000**
Car	0.778	0.400	0.566	*0.000*	0.896	0.937	**0.984**
CBF	0.858	*0.758*	0.909	0.870	0.994	0.991	**0.997**

The instances in bold text depicts best, and italics depicts worst performances

The experimental results indicate that FCN is a strong baseline for multivariate TSC and our proposed model Att-dGRU-SE-FCN outperformed over present state-of-the-art methods, MLSTM-FCN & MALSTM-FCN, and FCN in terms of classification testing error and *f1-score*.

5.3 Ablation Study

The ablation study is provided to determine the impact of each module of our proposed models. Table 7 illustrates the performance of the univariate model Att-dGRU-FCN on

Table 8 The ablation study on Att-dGRU-SE-FCN model

Datasets	GRU	Att-GRU	dGRU	Att-dGRU	FCN	SE-FCN	dGRU-SE-FCN	Att-dGRU-SE-FCN
Testing error rate								
EEG	*0.531*	0.453	*0.531*	0.531	0.500	**0.437**	0.453	**0.437**
HT Sensor	*0.560*	0.300	0.339	0.300	0.380	0.360	0.380	**0.140**
ArabicDigits	*0.272*	0.124	0.095	0.087	0.007	0.005	0.008	**0.004**
ECG	0.129	0.159	0.139	0.149	0.149	0.149	*0.199*	**0.110**
LP1	*0.400*	0.380	0.380	0.300	0.180	0.160	0.180	**0.140**
F1-score								
EEG	*0.469*	0.547	0.469	0.469	0.500	**0.563**	0.547	**0.563**
HT Sensor	*0.161*	0.714	0.667	0.707	0.626	0.612	0.620	**0.860**
ArabicDigits	*0.727*	0.877	0.906	0.913	0.992	0.995	0.992	**0.996**
ECG	0.870	0.840	0.860	0.850	0.850	0.850	*0.800*	**0.890**
LP1	*0.600*	0.626	0.620	0.700	0.820	0.840	0.828	**0.860**

The instances in bold text depicts best, and red depicts worst performances some datasets with modules, while Table 8 demonstrates the performance of our multivariate model Att-dGRU-SE-FCN on a few datasets. The datasets used in our study is extensive, so we have chosen random datasets to perform an ablation study.

In Table 7, our model Att-dGRU-FCN showed the best performance over all the modules. GRU and Att-GRU depicted the worst results, while dGRU and Att-dGRU performed slightly better than GRU & Att-dGRU. The dGRU-FCN's performance is better than FCN.

In Table 8, the Att-dGRU-SE-FCN model outperformed on all modules. SE-FCN depicts a similar performance on the EEG dataset along with Att-dGRU-SE-FCN. We noticed that Att-dGRU is slightly better than Att-GRU, and dGRU also depicted superior results than GRU for these multivariate tasks, excluding the ECG dataset. On the ECG dataset, GRU's performance is higher than dGRU, while on the EEG dataset, GRU & dGRU showed identical results. SE-FCN showed significant performance than FCN and dGRU-FCN.

Figures 10 and 11 presents the training and validation testing error loss on each module of the Att-dGRU-FCN & Att-dGRU-SE-FCN models over the Beetlefly and HT sensor datasets. It is evident that generalization gets better when we add more modules to the model, and our models, Att-dGRU-FCN & Att-dGRU-SE-FCN graphs, generate the smoothest curves amongst all of their modules graphs.

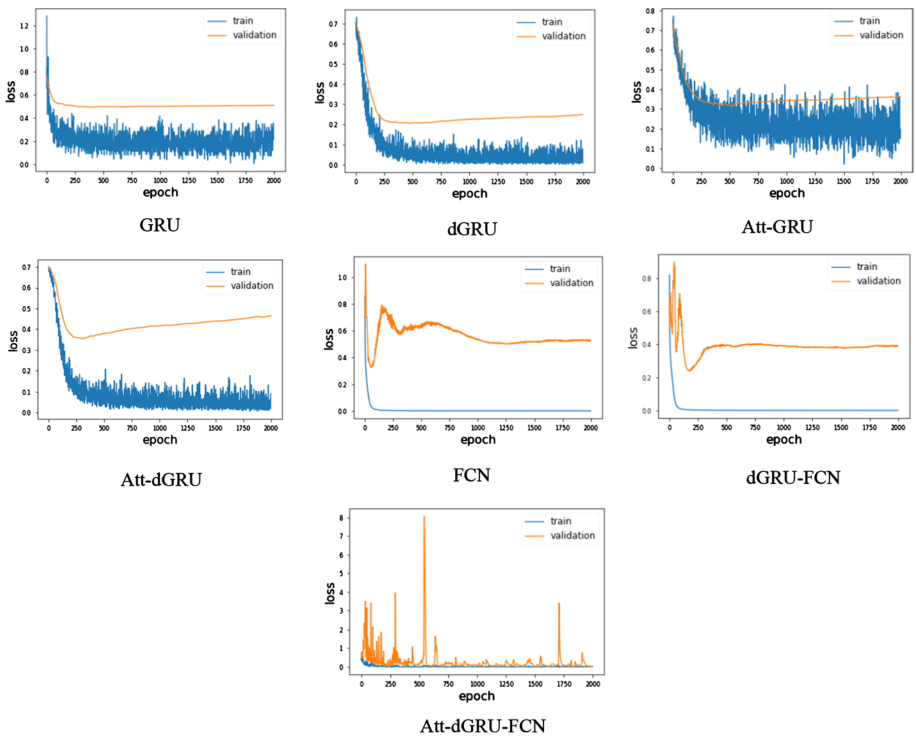


Fig. 10 The ablation study showing training and validation testing error loss on each module of Att-dGRU-FCN model over Beetlefly dataset

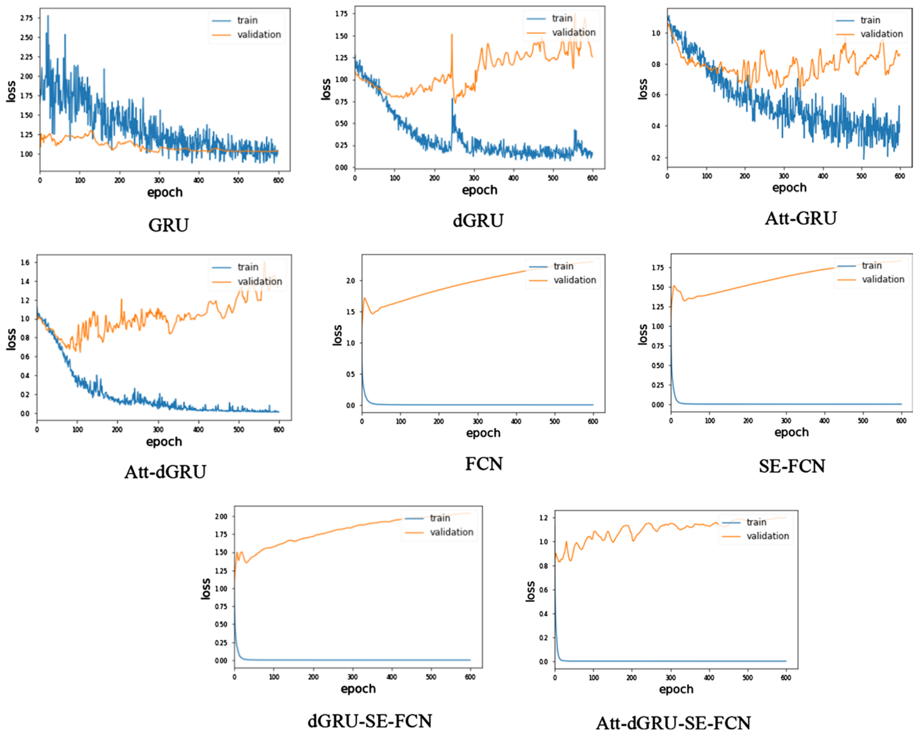


Fig. 11 The ablation study showing training and validation testing error loss on each module of Att-dGRU-SE-FCN model over HT sensor dataset

In general, our experiments carried out on more than a hundred datasets enabled us to validate the performance of our contributions. Our proposed methods are novel and efficient and can perform significantly better than the state-of-the-art techniques and baselines without requiring any heavy data pre-processing, feature crafting, refining, or fine-tuning.

6 Conclusions

This paper studies the problem of univariate and multivariate TSC by introducing two hybrid end-to-end deep learning models. The main idea of this study is to exploit the attention mechanism, dGRU, FCN, and SE block in hybrid deep neural networks for proficient performance. The proposed models are validated on multiple univariate and multivariate benchmark datasets, and the results indicate that these models depict significantly better performance than state-of-the-art methods and the baselines. The experimental results also prove that these models can classify time series more accurately than many well-known published methods by exploiting attention mechanism, dGRU, SE block, and FCN in hybrid deep neural networks.

Acknowledgements This paper was partially supported by NSFC Grant U1509216, U1866602, 61602129, and Microsoft Research Asia.

References

1. Zhang J, Li Y, Xiao W, Zhang Z (2020) Non-iterative and fast deep learning: multilayer extreme learning machines. J Frankl Inst 357:8925–8955
2. Zhang J, Xiao W, Li Y, Zhang S, Zhang Z (2020) Multilayer probability extreme learning machine for device-free localization. Neurocomputing 396:383–393
3. Zhang J, Xiao W, Li Y, Zhang S (2018) Residual compensation extreme learning machine for regression. Neurocomputing 311:126–136
4. Aswolinskiy W, Reinhart RF, Steil J (2018) Time series classification in reservoir-and model-space. Neural Process Lett 48:789–809
5. Keogh E, Ratanamahatana CA (2005) Exact indexing of dynamic time warping. Knowl Inf Syst 7:358–386
6. Lin J, Keogh E, Wei L, Lonardi S (2007) Experiencing SAX: a novel symbolic representation of time series. Data Min Knowl Discov 15:107–144
7. Baydogan MG, Runger G, Tuv E (2013) A bag-of-features framework to classify time series. IEEE Trans Pattern Anal Mach Intell 35:2796–2802
8. Schäfer P (2015) The BOSS is concerned with time series classification in the presence of noise. Data Min Knowl Discov 29:1505–1530
9. Schäfer P (2016) Scalable time series classification. Data Min Knowl Discov 30:1273–1298
10. Schäfer P, Leser U (2017) Fast and accurate time series classification with weasel. In: Proceedings of the 2017 ACM on conference on information and knowledge management, pp 637–646
11. Lines J, Bagnall A (2015) Time series classification with ensembles of elastic distance measures. Data Min Knowl Discov 29:565–592
12. Bagnall A, Lines J, Hills J, Bostrom A (2015) Time-series classification with COTE: the collective of transformation-based ensembles. IEEE Trans Knowl Data Eng 27:2522–2535
13. Lines J, Taylor S, Bagnall A (2016) 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
14. Lines J, Taylor S, Bagnall A (2018) Time series classification with HIVE-COTE: the hierarchical vote collective of transformation-based ensembles. ACM Trans Knowl Discov Data. https://doi.org/10.1145/3182382
15. Shifaz A, Pelletier C, Petitjean F et al (2020) TS-CHIEF: a scalable and accurate forest algorithm for time series classification. Data Min Knowl Disc 34:742–775. https://doi.org/10.1007/s10618-020-00679-8
16. Zheng Q, Yang M, Yang J, Zhang Q, Zhang X (2018) Improvement of generalization ability of deep CNN via implicit regularization in two-stage training process. IEEE Access 6:15844–15869
17. Zheng Q, Zhao P, Li Y et al (2020) Spectrum interference-based two-level data augmentation method in deep learning for automatic modulation classification. Neural Comput Appl. https://doi.org/10.1007/s00521-020-05514-1
18. Zheng Q, Tian X, Yang M et al (2020) PAC-Bayesian framework based drop-path method for 2D discriminative convolutional network pruning. Multidimens Syst Signal Process 31:793–827. https://doi.org/10.1007/s11045-019-00686-z
19. Zheng Q, Tian X, Jiang N, Yang M (2019) Layer-wise learning based stochastic gradient descent method for the optimization of deep convolutional neural network. J Intell Fuzzy Syst 37:5641–5654
20. Zheng Q, Yang M, Tian X, Jiang N, Wang D (2020) A full stage data augmentation method in deep convolutional neural network for natural image classification. Discrete Dyn Nat Soc. https://doi.org/10.1155/2020/4706576
21. Khan M, Wang H, Ngueilbaye A et al (2020) End-to-end multivariate time series classification via hybrid deep learning architectures. Pers Ubiquit Comput. https://doi.org/10.1007/s00779-020-01447-7
22. Khan M, Wang H, Riaz A et al (2021) Bidirectional LSTM-RNN-based hybrid deep learning frameworks for univariate time series classification. J Supercomput. https://doi.org/10.1007/s11227-020-03560-z
23. Wang Z, Yan W, Oates T (2017) Time series classification from scratch with deep neural networks: a strong baseline. In: 2017 international joint conference on neural networks (IJCNN), pp 1578-–1585
24. Karim F, Majumdar S, Darabi H, Chen S (2018) LSTM fully convolutional networks for time series classification. IEEE Access 6:1662–1669
25. Elsayed N, Maida AS, Bayoumi M (2018) Deep gated recurrent and convolutional network hybrid model for univariate time series classification. arXiv:1812.07683

26. Fawaz HI, Lucas B, Forestier G, Pelletier C, Schmidt DF, Weber J et al (2020) Inceptiontime: finding alexnet for time series classification. Data Min Knowl Discov 34:1936–1962
27. Dempster A, Petitjean F, Webb GI (2020) ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. Data Min Knowl Discov 34:1454–1495
28. Seto S, Zhang W, Zhou Y (2015) Multivariate time series classification using dynamic time warping template selection for human activity recognition. In: 2015 IEEE symposium series on computational intelligence, pp 1399-–1406
29. Schäfer P, Leser U (2017) Multivariate time series classification with WEASEL+ MUSE. arXiv:1711.11343
30. Baydogan MG, Runger G (2015) Learning a symbolic representation for multivariate time series classification. Data Min Knowl Discov 29:400–422
31. Zheng Y, Liu Q, Chen E, Ge Y, Zhao JL (2014) Time series classification using multi-channels deep convolutional neural networks. In: International conference on web-age information management, pp 298–310
32. Karim F, Majumdar S, Darabi H, Harford S (2019) Multivariate lstm-fcns for time series classification. Neural Netw 116:237–245
33. Zhang X, Gao Y, Lin J, Lu C-T (2020) TapNet: multivariate time series classification with attentional prototypical network. In: AAAI, pp 6845–6852
34. Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. arXiv:1409.0473
35. Wang N, Ma S, Li J, Zhang Y, Zhang L (2020) Multistage attention network for image inpainting. Pattern Recogn 106: https://doi.org/10.1016/j.patcog.2020.107448
36. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv:1502.03167
37. Nair V, Hinton GE (2010) Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10), pp 807–814
38. Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555
39. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15:1929–1958
40. He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision, pp 1026–1034
41. Lin M, Chen Q, Yan S (2013) Network in network. arXiv:1312.4400
42. Chollet F et al (2015). Keras. GitHub. Retrieved from https://github.com/fchollet/keras
43. Hu J, Shen L, Sun G (2018) Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7132–7141
44. Chen Y, Keogh E, Hu B, Begum N, Bagnall A, Mueen A, Batista G (2015) The UCR time series classification archive. http://www.cs.ucr.edu/~eamonn/time_series_data/
45. Pei W, Dibeklioğlu H, Tax DM, van der Maaten L (2017) Multivariate time-series classification using the hidden-unit logistic model. IEEE Trans Neural Netw Learn Syst 29:920–931
46. Dua D, Graff C (2017) UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences
47. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv:1412.6980
48. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J et al (2016) Tensorflow: a system for large-scale machine learning. In: 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16), pp 265–283