TRAFFIC DATA ANALYSIS USING DEEP ELMAN AND GATED RECURRENT AUTO-ENCODER

S. Mehralian[*], *M. Teshnehlab*[†], *B. Nasersharif*[‡]

Abstract: Traffic flow prediction is one of the most interesting machine learning applications in real-world problems that can help anyone move around. In this study, we proposed a feature extraction structure for multivariate time series using Elman recurrent auto-encoder. We added loopback from the encoder layer of the normal auto-encoder to regard sequence information between successive data. The feedback layer implemented using Elman neural network and GRU cells, then the model is trained by different optimization algorithms. The models are also trained using the Emotional Learning method in which we involve the derivative of the error in the cost function to avoid local minimums and keep the last state of the network. We used the proposed method for classification and prediction problems on traffic data from the California Department of Transportation Performance Measurement System (PeMS). The results show that our structure can successfully extract a compact representation of traffic data useful for reconstructing of original data, classification, and prediction. The results also show that adding the recurrent layer to the feature extractor (auto-encoder) leads to better results in the classification phase in comparison with standard methods that do not use the recurrence during feature extraction.

Key words: *deep learning, recurrent neural network, auto-encoder, traffic data analysis*

Received: August 4, 2019 **DOI:** 10.14311/NNW.2020.30.023
Revised and accepted: December 30, 2020

1. Introduction

Today, monitoring and analyzing traffic data are integral parts of every Intelligent Transportation System (ITS), in which traffic information such as flow, occupancy, and speed are stored and processed to have better insight into the transportation systems. The ITSs are critical, especially in cities with a massive population and

[*]Soheil Mehralian; Computer Engineering Faculty, K.N. Toosi University of Technology, Tehran, Iran, E-mail: mehralian@ee.kntu.ac.ir

[†]Mohammad Teshnehlab – Corresponding author; Industrial Control Center of Excellence, K.N. Toosi University of Technology, Tehran, Iran, E-mail: teshnehlab@eetd.kntu.ac.ir

[‡]Babak Nasersharif; Computer Engineering Faculty, K.N. Toosi University of Technology, Tehran, Iran, E-mail: bnasersharif@kntu.ac.ir

many vehicles, to avoid the problem such as congestion, accidents, and air pollution. One of the advanced problems in this field is predicting future traffic; this is one of the critical information that everyone needs because information such as current state and prediction for a few hours later can help us choose better paths and have a clearer sky.

Historical information on the traffic was needed to predict traffic flow; they can be gathered from different sources such as radars, cameras, mobile global positioning systems, crowdsourcing social media, wireless magnetic sensors [6, 11]. So, backed by the availability of exploding traffic data, the development of data-driven traffic prediction methods becomes more popular and efficient. These data can be used for discovering patterns and clustering, description and prediction of traffic, and planning and recommendation for new routes. In addition to the analyses mentioned above, the data acquired from traffic data also can be used for visualization purposes to get a more in-depth insight into them [5].

Nonlinear intelligent algorithms were used for traffic data analysis and behavior investigation because traffic systems are complex systems.

In this study, a new method based on deep learning is developed and applied to the traffic flow data. The proposed method is feature extraction and dimensionality reduction method for multivariate time-series and based on an Auto-Encoder. Learned features of the proposed method applied to traffic flow data in two case studies: in the first one, learned features in a classification problem were used, i.e., classifying time-series into corresponding weekdays from which data gathered, and in the second one is a multivariate time-series prediction problem in which it has been tried to predict short-term traffic flow using available historical data.

After the introduction, the rest of the paper is organized as follows: Section 2 reviews some related works about the traffic flow prediction and analysis using intelligent algorithms. Section 3 contains some preliminaries about deep learning, auto-encoders, and recurrent neural networks needed to understand the rest of the paper. The proposed method is presented in Section 4, results, dataset, and case studies explained in 5, and finally, the paper concluded in Section 6 of the text.

2. Related works

During the last few decades, traffic becomes an inevitable problem in urban management systems, and many pieces of research using the different methods have been conducted to solve the traffic problem. The methods used to deal with traffic data can be classified into three categories: prediction, interpolation, and statistical learning [22].

Prediction methods use historical data to build models for future or missing data prediction. One of the commonly used methods of this type is the Auto-Regressive Integrated Moving Average (ARIMA) [20, 25]. During the 2000s, several versions of ARIMA are proposed and used as a prediction tool for traffic flow data, such as ARIMAX [28], ARMA and space-time ARIMA [18], seasonal ARIMA [29].

Interpolation methods, or history models, use neighboring data or historical data points to replace missing or corrupted data. Some simple methods, such as filling missing data at the same time from previous days, are of this type [1]. As another method that uses neighboring data points to interpolate missing data point in traffic flow, k-NN based methods can be mentioned [3]. Nonparametric methods such as k-NN attracted many researchers in traffic flow prediction because they can handle nonlinear and stochastic traffic flow behavior more conveniently [10, 12].

Statistical learning methods use historical data to build a model to inference missing or future data points. They reach the solution via solving an iterative optimization problem in which a cost function will be minimized. Methods based on neural networks [23] are the most common methods of this type. The proposed method in this study is also a statistical learning method based on a recurrent neural network for feature extraction.

With the success of some simple prediction methods, gradually more advanced intelligent methods were welcomed in this field, methods such as Bayesian Network [26, 31], Support Vector Regression (SVR) [17], Support Vector Machine (SVM) [30] and Artificial Neural Networks [19]. Methods such as SVM and ANN are nonlinear, these nonlinear intelligent algorithms were used for traffic data analysis and behavior investigation because traffic systems are complex systems [27], and a linear model cannot catch all nonlinearity of the system. In addition to analysis purposes, an intelligent algorithm is applicable in monitoring scenarios such as in [2], and the research was done by Lewandowski et al. based on information gathered using low energy devices, namely Bluetooth beacons [21].

After the advent of deep learning methods, they have been used widely in traffic data analysis, like many other applications; in the last part of related works, some recent works are reviewed. In [34], factors such as weather and accident are also considered to predict traffic flow. In [36], LSTM and GRU are used to predict traffic flow in their vanilla form. The study in [33] used deep learning methods to tackle Spatio-temporal elements that affect the traffic, such as sharp nonlinearities in traffic flow behavior because of transitions between free flow, breakdown, recovery, and congestion. In [35], traffic data analysis conducted using big data and DNN based traffic flow (BTF); they also used an attention-based model to wight past data importance. Another attention-based method is [37], in which the Conv-LSTM network used for short term traffic flow prediction. The work of Zhang et al. [38] is also is used deep convolution neural networks to extract Spatio-temporal features from data and predict the short-term flow.

3. Deep Learning and RNNs: Preliminaries

During the last decade, Deep Learning methods were the most attractive field in artificial intelligence; backed by the artificial neural network, deep neural networks are more similar to our brain than ever before. Among different phases of a machine learning process, the phase in which deep method can positively contribute is the feature extraction phase of machine learning application; in other words, instead of selecting/extracting features manually, deep learning methods can be used to extract the best features for the data on the hand.

In the next subsections of the paper, some descriptions about deep learning methods, auto-encoders, and GRU networks, is presented.

3.1 Auto-Encoders

Auto-Encoders are unsupervised learning methods for learning representations: extracting features representing the original data. These features also can reproduce the data with a minimum reconstruction error. Auto-encoders compress data in the case of a lower dimension for extracted features; thus, they learn Compact Representation of the original data.

Fig. 1 shows the structure of an auto-encoder. It has two main parts: encoder and decoder. The encoder projects data into latent space in which data should be reconstructed with minimum reconstruction error using the decoder. i.e., the following equation holds about input and output of auto-encoder:

$$input = decoder(encoder(input)). \tag{1}$$

The feature learned (the Coded version of input) is the bottleneck (hidden) layer output. The auto-encoder will find any data structure and the correlation between features through this mechanism and extract new features and compress the data. The more dimensionality reduction can be acquired by stacking several encoders together.

The auto-encoder structure has been discussed above lacks any solution to consider recurrency and sequence information of samples. Thus, in this paper, we have added a recurrent layer to it to do so.

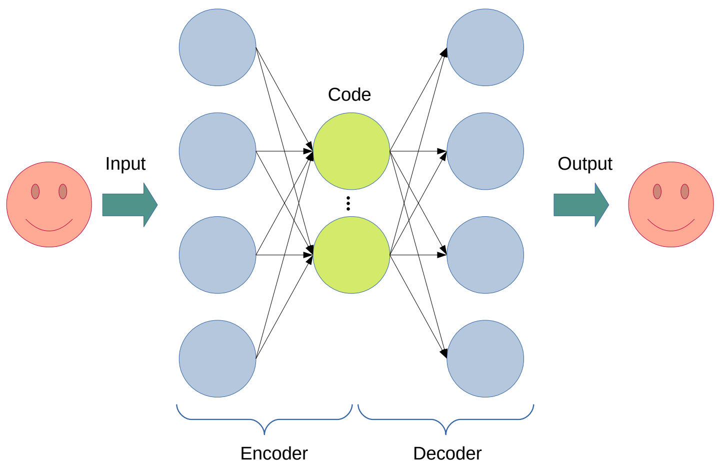


Fig. 1 *A simple structure of auto-encoder.*

3.2 Recurrent Neural Networks

Recurrent neural networks (RNNs), unlike feed-forward networks, can hold their internal state (memory) and handle temporal information when dealing with sequential data, it makes RNNs suitable for applications such as Language Modeling and Prediction [24], Speech Recognition [14], Machine Translation [4], and Image Recognition and Characterization [32]. RNNs are also very useful for dealing with time-variant systems. It loops back the current output into the network; therefore, it has two inputs: the current input and the previous time step's output. This mechanism results in the extraction of information in the data sequence, i.e., information about what is happening next.

As another difference, RNN can map one-to-many, many-to-many, and many-to-one, while feed-forward neural networks only map one input to one output.

Standard RNNs suffer from gradient vanishing problem, which is an issue during the training phase of the neural networks in which gradient-based optimization methods and Backpropagation is used; in these algorithms, the gradient becomes vanishingly small for layers near the first layer of the network. Cho et al. [7] proposed the GRU neural network, a variation of LSTM networks, to eliminate this problem.

The GRU networks solve the gradient vanishing problem using its gates: *Update Gate and Reset Gate*. They are responsible for deciding which information should be passed to the output of the cell. This network's training process consists of training these gates to hold information from a long time ago and forgetting them.

Each GRU network cell has two inputs: x_t and h_{t-1}, which are current input, at timestamp t, and output of the previous time-step $(t-1)$, receptively. The following equation can be used to formulate the behavior of GRU networks for updates and reset gates. The output of the update gate is as follows:

$$z_t = \sigma(W^{(z)} x_t + U^{(z)} h_{t-1}). \tag{2}$$

In which $W^{(z)}$ and $U^{(z)}$ are weight matrices for current and previous step inputs, respectively. The formula in (2) determines how much information from the previous step should be passed to the future steps.

The output of reset is also as following:

$$r_t = \sigma(W^{(r)} x_t + U^{(r)} h_{t-1}). \tag{3}$$

In which $W^{(r)}$ and $U^{(r)}$ are weight matrices for current and previous step inputs, respectively, nearly the same as (2). In both equations above (2 and 3), $\sigma(\cdot)$ is the sigmoid function used as the unit's activation function. The gate defined in (3) decides how much information should be forgotten.

Regarding the gates defined in (2) and (3), the current memory context can be calculated as the equation below

$$h'_t = \tanh(W x_t + r_t \odot U h_{t-1}). \tag{4}$$

Thus, the final memory for the current step will be as follows:

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h'_t. \tag{5}$$

Now, the output in (5) can be used as an input of the next time step and repeat Eqs. (2) through (5) for that time.

4. Algorithm of the proposed method

In this section, the overview and details of the proposed method are presented. This method is based on normal auto-encoder, which its main application is feature learning and dimensionality reduction. We will also use it for this purpose, but it is enhanced with a recurrent layer, as in Elman neural networks [13]. The idea behind using the recurrent layer in the model is to involve time and sequence information in the feature extraction process. To do so, as Fig. 2 shows, the network has a feedback loop from the hidden layer to the input layer; it helps the auto-encoder to deal more efficiently with sequential and time-series data.

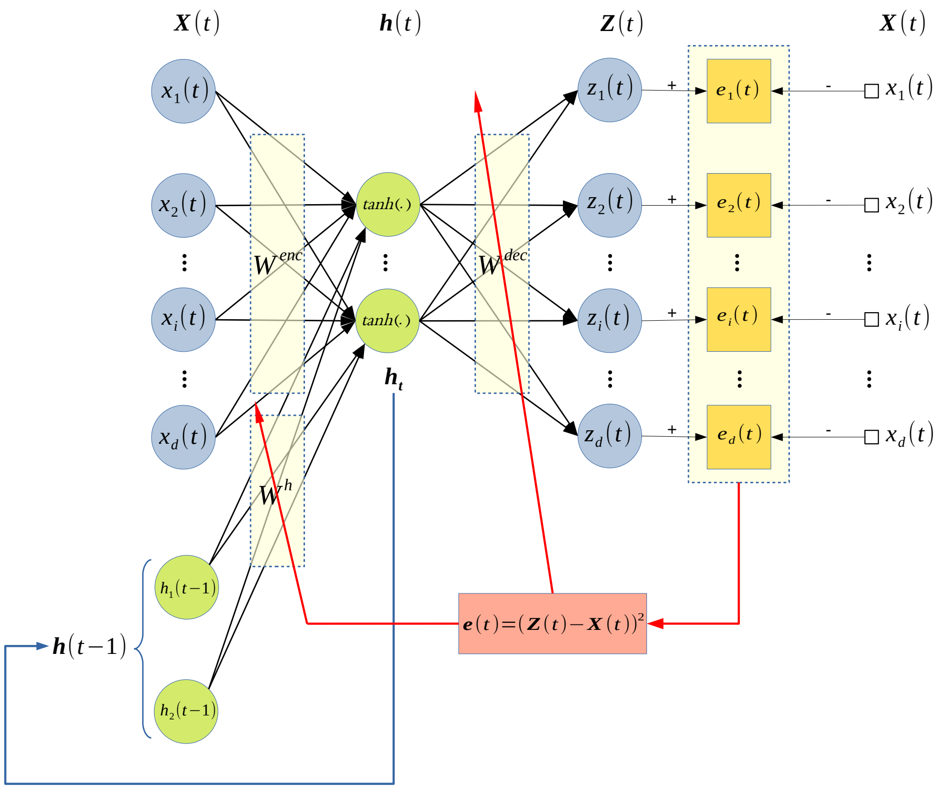


Fig. 2 *The structure of the proposed model formulated above. Red lines are error backpropagation, blue lines are feedback paths, and black ones are feed-forward paths.*

Feed-forward pass of Recurrent Auto-Encoder (RAE) algorithm can be formulated as:

$$net^{(enc)} = \mathbf{X}(t)\mathbf{W}^{(enc)} + b^{(enc)}, \tag{6}$$

$$net^{(h)} = \mathbf{h}(t-1)\mathbf{W}^{(h)}, \tag{7}$$

$$\mathbf{h}(t) = f_1(net^{(enc)} + net^{(h)}), \tag{8}$$

$$net^{(dec)} = \mathbf{h}(t)\mathbf{W}^{(dec)} + b^{(dec)}, \tag{9}$$

$$\mathbf{Z}(t) = f_2(net^{(dec)}), \tag{10}$$

in which $f_i(\cdot)$ for $i = 1, 2$ can be $\tanh(\cdot)$ function. However, if a linear activation function in the output layer (decoder) is preferred, you can choose a linear function such as the Identity function for $f_2(\cdot)$, $\mathbf{X}(t)$ is d-dimensional input of the network at time step t, $\mathbf{W}, b$ are network parameters, and $\mathbf{Z}(t)$ is the output of the network at time step t.

The cost function of the network mentioned above tries to minimize the reconstruction error using Mean Squared Error (MSE) defined below:

$$J(t) = \frac{1}{2n} \sum_{i=1}^{n} \mathbf{e}_i(t)^2, \tag{11}$$

where

$$\mathbf{e}_i(t) = (\mathbf{X}_i(t) - \mathbf{Z}_i(t)) \tag{12}$$

and n is the number of training samples. Regarding the Eq. (11), the derivatives and update equations of the network's cost function can be calculated respect to weights and biases as following:

$$\Delta \mathbf{W}^{(dec)}(t) = -\eta \frac{\partial J}{\partial \mathbf{W}^{(dec)}}(t). \tag{13}$$

In which η is the learning rate. The right side of the equation above is calculated as below:

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{W}^{(dec)}}(t) &= \frac{\partial J}{\partial e} \frac{\partial e}{\partial z} \frac{\partial z}{\partial net^{(dec)}} \frac{\partial net^{(dec)}}{\partial \mathbf{W}^{(dec)}}(t) \\ &= e(t)(-1) f_2'(net^{(dec)}) h_t(t) \\ &= -e(t) f_2'(net^{(dec)}) h_t(t). \end{aligned} \tag{14}$$

So, regarding equations (13) and (14):

$$\Delta \mathbf{W}^{(dec)}(t) = \eta e(t) f_2'(net^{(dec)}) h_t(t). \tag{15}$$

For bias:

$$\Delta b^{(dec)}(t) = -\eta \frac{\partial J}{\partial b^{(dec)}}(t), \tag{16}$$

in which $\frac{\partial J}{\partial b^{(dec)}}(t)$ is calculated as the equation below:

$$\begin{aligned} \frac{\partial J}{\partial b^{(dec)}}(t) &= \frac{\partial J}{\partial e} \frac{\partial e}{\partial z} \frac{\partial z}{\partial net^{(dec)}} \frac{\partial net^{(dec)}}{\partial b^{(dec)}}(t) \\ &= e(t)(-1) f_2'(net^{(dec)})(1)(t) \\ &= -e(t) f_2'(net^{(dec)})(t). \end{aligned} \tag{17}$$

According to equations (16) and (17):

$$\Delta b^{(dec)}(t) = \eta e(t) f_2'(net^{(dec)})(t). \tag{18}$$

For the sake of simplicity and convenience in network training, $\mathbf{W}^{(enc)} = transpose$ $(\mathbf{W}^{(dec)})$; thus, only the output layer weights were trained except for $\mathbf{W}^h$ for which there is no equivalent weight matrix in the output layer. The derivative of this weight matrix can be calculated as follows:

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{W}^{(h)}}(t) &= \frac{\partial J}{\partial e}\frac{\partial e}{\partial z}\frac{\partial z}{\partial net^{(dec)}}\frac{\partial net^{(dec)}}{\partial h_t}\frac{\partial h_t}{\partial net^{(h)}}\frac{\partial net^{(h)}}{\partial \mathbf{W}^{(h)}}(t)\\ &= -e(t)f_2'(net^{(dec)})\mathbf{W}^{(dec)}f_1'(net^{(dec)}+net^{(h)})\frac{\partial net^{(h)}}{\partial \mathbf{W}^{(h)}}(t).\end{aligned} \tag{19}$$

In the last part of Eq. (19), in addition to net^h, $\mathbf{W}^h$ can be seen in h_t, h_{t-1}, h_{t-2}, and so forth. Thus, there is a recursion on calculating the calculation of the derivative of $\mathbf{W}^h$ that makes derivation calculation a bit tricky. In situations like this, the BackPropagation Through Time (BPTT) rule will be used in which the derivative of variables like $\mathbf{W}^h$ are estimated through limited steps in time, so for the last step of (19), there is:

$$\begin{aligned}\frac{\partial net^{(h)}}{\partial \mathbf{W}^{(h)}}(t) &= h(t-1)+\frac{\partial h(t-1)}{\partial \mathbf{W}^{(h)}}\\ &\approx h(t-1)+\frac{\partial h(t)}{\partial \mathbf{W}^{(h)}}\\ &= h(t-1)+\frac{\partial h(t)}{\partial net^{(h)}}\frac{\partial net^{(h)}}{\partial \mathbf{W}^{(h)}}\\ &= h(t-1)+f_1'(net^{(dec)}+net^{(h)})h(t-1).\end{aligned} \tag{20}$$

Following Eqs. (19) and (20), the update rule for $\mathbf{W}^{(h)}$ will be

$$\begin{aligned}\Delta\mathbf{W}^{(h)}(t) = \eta e(t)f_2'(net^{(dec)})\mathbf{W}^{(dec)}f_1'(net^{(dec)}+net^{(h)})\\ \times\left(h(t-1)+f_1'(net^{(dec)}+net^{(h)})h(t-1)\right).\end{aligned} \tag{21}$$

The cost function above will be minimized using Gradient Descent as in Eq. (22), and the Adam Optimizer method; the results and comparison of these methods will be described in Section 5 of the paper.

$$\theta_{new} = \theta_{old} - \eta\nabla J(\theta). \tag{22}$$

In Eq. (22), θ is the updatable parameter (e.g. $b^{(i)}$ and $\mathbf{W}^{(i)}$ for $i = 1, 2$) and η, as mentioned previously, is the learning rate which is $0 < \eta < 1$.

4.1 Emotional Learning

Also, the model presented in this study has been trained using the Emotional Learning method [16], in which the error of the t-th step of the network is calculated using the equation below:

$$J(t) = \frac{1}{2}\sum_{i=1}^{n}\left(k_1 e_i(t) + k_2\dot{e}_i(t)\right)^2. \tag{23}$$

In (23), k_1 and k_2 are weights that were assigned to the error $e(\cdot)$ and its derivative $\dot{e}(\cdot)$ for the sake of the trade-off between them. The error function $e(\cdot)$ is defined as Eq. (12). Thus, according to Eq. (12) and (23), the cost function of the Emotional Auto-Encoder can be rewritten in the following form:

$$J_{AE_{\text{Emo}}}(t) = \frac{1}{2}\sum_{i=1}^{n}\left((k_1 + k_2)e_i(t) - k_2 e_i(t-1)\right)^2. \tag{24}$$

The idea behind the training network using the emotional method is to avoid local minimums by invoking the previous step's error in the form of its derivative in the cost function.

5. Results and analysis

This section of the paper presents results of case studies used for performance evaluation of the proposed method. Two case studies have been used: classification and multivariate time-series prediction, but before them, the experiments' dataset is introduced briefly.

5.1 Dataset

In this work, PeMS-SF [9] is used as a benchmark dataset. This dataset consists of 15 months (January 1, 2008, to March 30, 2009) daily traffic data downloaded from the California Department of Transportation PEMS website[1]. These data sampled every 10 minutes using 963 stations from San Francisco freeways. Thus, there is a Time-series with a length of 144 (6×24) and the dimension of 963 for every day. Some prepossessing and cleaning is applied to data, for instance, holidays removed and all days from which data sampled are normal days; this results in a dataset with 440 time-series labeled with $\{1, \ldots, 7\}$ which represents Monday to Sunday, respectively. In the experiments, 267 data for the training phase and 173 out of 440 for test and performance evaluation were used.

5.2 Use Case 1: Classification

In the previous section of the paper, the PeMS-FS dataset is labeled by weekdays from which traffic data gathered. Thus, the first problem which should be solved is a classification problem using these data, i.e., the original task proposed with the dataset is to classify the data instances as the correct day of the week. The experiments in this section are conducted using the structure in Fig. 3; the figure shows train and test data are projected using learned features from the feature learner box (Fig. 2) and then fed to the classifier separately for train and evaluation

In the experiment, the first 100 features of the dataset out of 963 were selected and their dimensionality reduced from 100 to 80, and finally to 60 dimensions using stacked networks of the proposed structure. The network's hidden layer's activation function is $\tanh(\cdot)$, and the output layer is linear. The learning rate also is 0.01 to 0.05, and the time-step is 144. Then, learned features classified using two different

[1]Available at: `http://pems.dot.ca.gov`

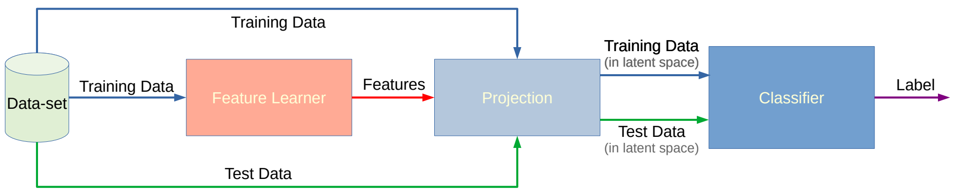


Fig. 3 *The scenario in which classification use case experiments conducted.*

classifiers: simple Multi-Layered Perceptron (MLP) and Recurrent Neural Network (RNN). Two optimizers were also used to train the presented models: Gradient Descent (GD) and Adam Optimizer (AO). Fig. 4 shows the proposed method's structure in the stacked mode that its output fed to the classifiers mentioned above. Tab. I shows the details of the performance of each method.

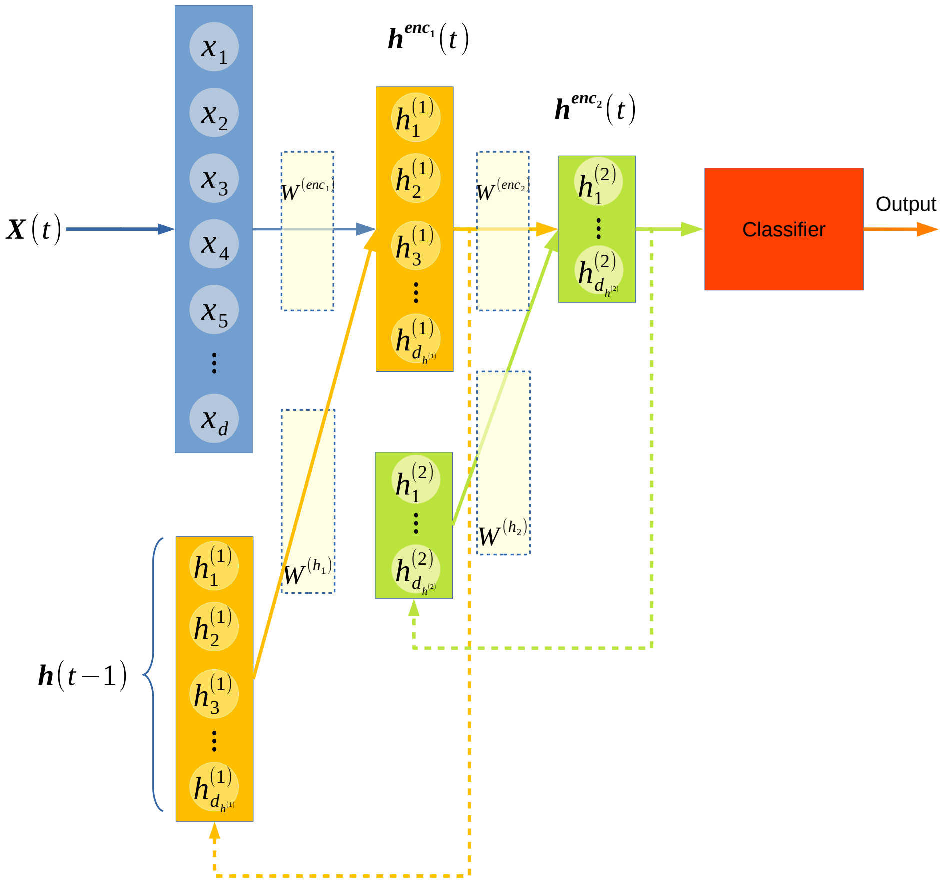


Fig. 4 *The proposed method in stacked mode. Dashed lines show feedback paths, and solid ones show forward paths.*

Despite the fewer dimensions and compact representation of features extracted using the proposed method, Tab. I shows that the presented approach has better classification performance regarding the average accuracy of five different and independent runs and their standard deviations. The table also indicates Adam Optimizer's superiority over the Gradient Descent training method.

	Only Classifier [100]	Our Method [100,80]	Our Method [100,80,60]	Our Method (Emo.) [100,80]	Our Method (Emo.) [100,80,60]
MLP/GD	75.14 ± 0.40	77.26 ± 2.03	77.46 ± 1.63	80.3 ± 0.73	78.93 ± 0.95
MLP/AO	76.81 ± 1.20	76.47 ± 1.21	75.72 ± 0.58	81.25 ± 0.64	78.17 ± 1.12
RNN/GD	55.10 ± 1.67	79.38 ± 0.67	67.62 ± 1.51	82.89 ± 0.46	80.40 ± 0.86
RNN/AO	76.59 ± 0.10	78.80 ± 0.67	76.68 ± 2.33	86.10 ± 0.42	83.22 ± 0.75

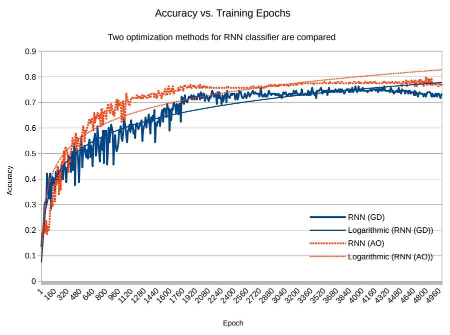
Tab. I *Classification accuracy of classifiers using different learning methods.*

Tab. II shows the confusion matrix for the classification of 173 test data using the recurrent neural network, optimized by Adam optimizer; the model trained using the Emotional Learning method. Fig. 8 depicts the emotional learning methods superiority compared with the normal learning process; as can be seen, the emotionally minimized loss for the proposed method is far better than the normal one. Fig. 5 also represents the accuracy of the classifiers (MLP and RNN), which learned using gradient descent and Adam Optimizer; during training, as you can see, RNN classifiers need more training epochs to reach their best results, but MLP classifiers in fewer epochs reach the best classification result. Fig. 6 depicts the performance of the same structure learned using emotional learning algorithm.

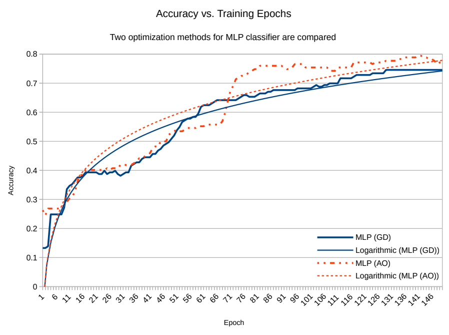
	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.
Mon.	28	0	0	0	0	0	0
Tue.	0	14	0	3	0	2	0
Wed.	0	2	8	4	0	0	0
Thu.	0	1	0	36	3	0	0
Fri.	0	0	0	5	15	2	0
Sat.	0	2	0	1	0	26	0
Sun.	0	0	0	0	0	0	22

Tab. II *Classification confusion matrix.*

The proposed approach compared with different methods, according to results reported in [39], the results depicted in Fig. 7, which shows the method in the current study, have better performance regarding the accuracy of classification. According to the figure and the results summarized in Tab. I, emotionally trained models have the best accuracy; it shows evolving the error of the previous steps can improve accuracy. Almost all methods in Fig. 7 can be used for classification and analysis of time series and have been applied to PeMS dataset. For instance, Multivariate Long Short-Term Memory Fully Convolutional Network (MLSTM-FCN) [41], Dynamic Time Warping (DTW) [40], and One Nearest Neighbor with Euclidean Distance (1NN-ED).

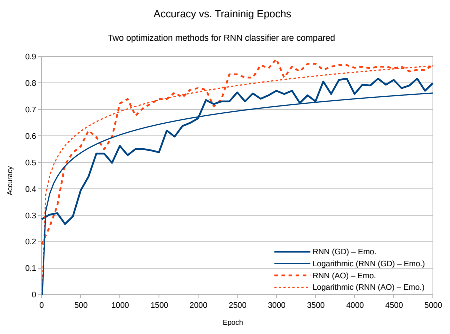


(a) *RNN Classifier*

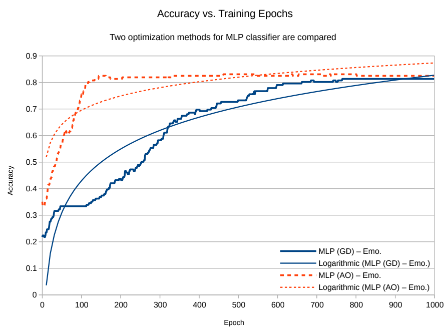


(b) *MLP Classifier*

Fig. 5 *Classification Accuracy on test data during training time. In (a) a Recurrent Neural Network used as the classifier and in (b) an MLP classifier is used, both classifiers optimized using Gradient Descent and Adam optimizer.*



(a) *RNN Classifier*



(b) *MLP Classifier*

Fig. 6 *Classification Accuracy on test data during training using the Emotional Learning method. In (a) a Recurrent Neural Network used as the classifier and in (b) an MLP classifier is used, both classifiers optimized using Gradient Descent and Adam optimizer.*

5.3 Use Case 2: Time-Series Prediction

In the second experiment, the proposed method has been used to predict traffic flow in short-time using historical data at the same place. The problem here is a type of multivariate time-series prediction. In this use case, the first 20 features of the PeMS-FS data is used as training samples, their dimensionality reduced from 20 to 15, 10, and finally to 5 using stacked auto-encoders, and finally a GRU Network [7] used to predict the 10-minute traffic flow in the future.

The structure used for feature extraction is the same as in Fig. 2, and its relation to the use case is similar to Fig. 3 except for the Classifier box; instead, in this experiment there is a time-series predictor, namely GRU Network. The GRU networks is selected for this experiments because they have better performance on small datasets [8] and fewer parameters compared with Long-Short Term Memory

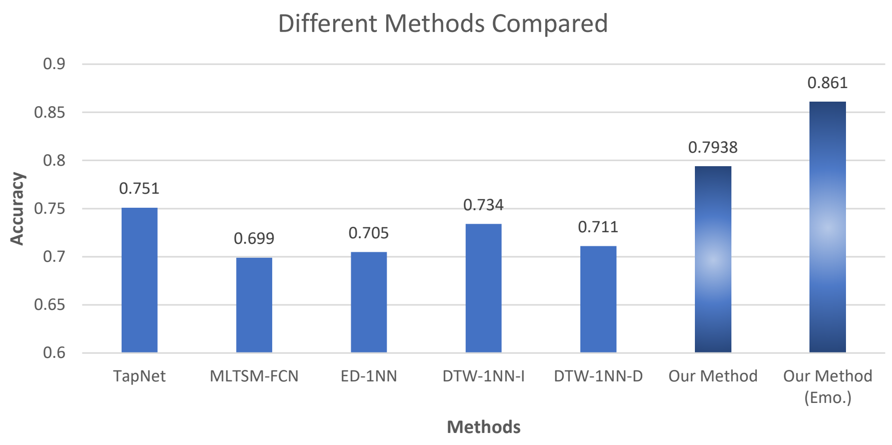


Fig. 7 *The classification accuracy of the proposed method compared with different methods on PeMS dataset.*

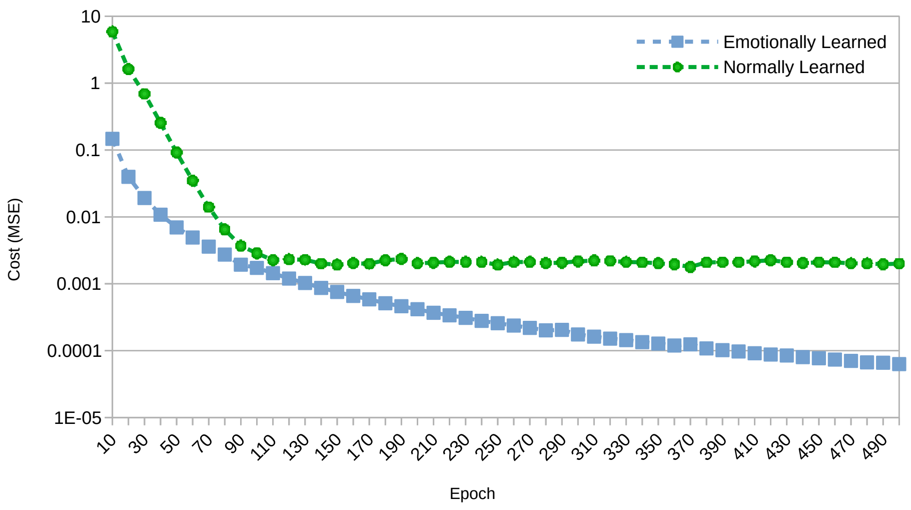


Fig. 8 *The loss of the minimization process of emotionally learned model and normally learned model of the proposed method compared during 500 epochs.*

(LSTM) networks [15]. The classifier has trained in batch mode with batch size of 100. Fig. 9 shows the network's loss function during the training phase on test data on a logarithmic scale; it is can be seen that the network successfully minimized the cost function defined for it.

Some sample predictions of traffic flow are illustrated in Fig. 10; the GRU network used for this experiment has a sequence length of 2, a learning rate of 0.001, 2 hidden layers, and 20 neurons in each layer. The network also uses an L2 loss function and regularization term with $\lambda = 0.0001$ as in equation below:

$$L_2 Loss(t) = \sum_{i=1}^{n} (y_{\text{desire}}(t) - y_{\text{prediction}}(t))^2 + \lambda \sum_{j=1}^{k} \mathbf{W}_j^2. \tag{25}$$

In (25), n is the number of training samples, k is the number of network layers, and $\mathbf{W}_j$ is the network's trainable parameters. The second term in (25) is the regularization term and used to avoid model over-fitting on training data.

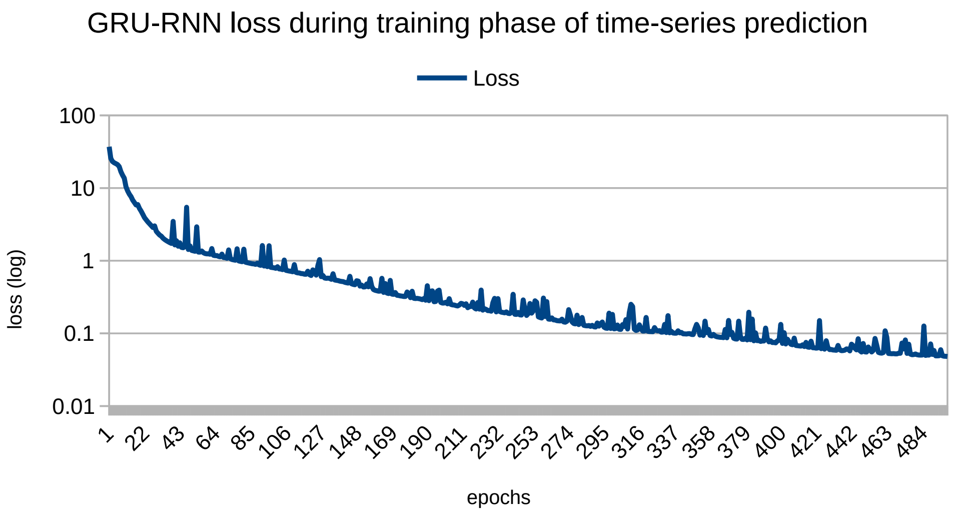


Fig. 9 *The loss for the second use case in which a GRU-RNN used to predict traffic flow.*

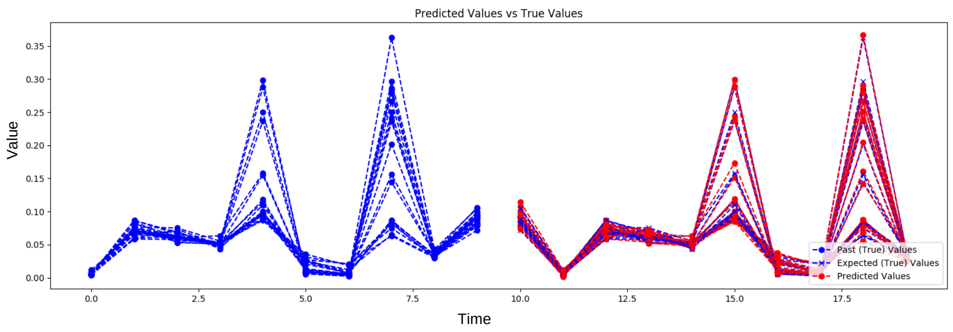


Fig. 10 *The loss for the second use case in which a GRU network used to predict traffic flow.*

6. Conclusion

In this paper, a feature extraction method for multivariate time-series is proposed. The proposed method is a deep model based on auto-encoder and recurrent methods. A recurrent layer has been added to the auto-encoder's hidden layer to empowers it to deal with time-series and extract time related information from raw data. The proposed model's recurrent layer was implemented by the Elman neural network and Gated Recurrent Units. Performance of the proposed method evaluated using two use cases: in the first one, learned features had been used in a classification problem; two classifiers, MLP and RNN, are trained using gradient descent and Adam optimizer algorithms, and used to classify learned features. Results show that in both classifiers, learned feature could help to reduce dimensionality and improve classification accuracy. Another learning strategy used in the study was the Emotional Learning method; the idea behind this strategy was to avoid local minima, which results in better accuracy in the classification use case. The results of this use case compared with different methods to show the superiority of our method. In the second use case, the features extracted using the proposed method helped us to predict traffic flow in a time-series prediction problem. The results of this experiment show that the proposed method successfully extracts relevant information from multivariate time-series.

References

[1] ALLISON P.D. *Missing data*, volume 136. Sage publications, 2001.

[2] BISWAS S.P., ROY P., PATRA N., MUKHERJEE A., DEY N. Intelligent traffic monitoring system. In: *Proceedings of the Second International Conference on Computer and Communication Technologies*, Springer, 2016, pp. 535–545.

[3] CHANG G., ZHANG Y., YAO D. Missing data imputation for traffic flow based on improved local least squares. *Tsinghua Science and Technology*, June 2012, 17(3), pp. 304–309.

[4] CHEN M.X., FIRAT O., BAPNA A., JOHNSON M., MACHEREY W., FOSTER G., JONES L., PARMAR N., SCHUSTER M., CHEN Z. The best of both worlds: Combining recent advances in neural machine translation. *arXiv preprint arXiv:1804.09849*, 2018.

[5] CHEN W., GUO F., WANG F. A survey of traffic data visualization. *IEEE Transactions on Intelligent Transportation Systems*, Dec 2015, 16(6), pp. 2970–2984.

[6] CHEUNG S.Y., ERGEN S.C., VARAIYA P. Traffic surveillance with wireless magnetic sensors. In *Proceedings of the 12th ITS world congress*, 2005, volume 1917, page 173181.

[7] CHO K., VAN MERRIËNBOER B., GULCEHRE C., BAHDANAU D., BOUGARES F., SCHWENK H., BENGIO Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[8] CHUNG J., GULCEHRE C., CHO K., BENGIO Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[9] CUTURI M. Caltrans, performance measurement system (pems), 2017.

[10] DAVIS G.A., NIHAN N.L. Nonparametric regression and short-term freeway traffic forecasting. *Journal of Transportation Engineering*, 1991, 117(2), pp. 178–188.

[11] DUAN Y., LV Y., LIU Y.L., WANG F.Y. An efficient realization of deep learning for traffic data imputation. *Transportation research part C: emerging technologies*, 2016, 72, pp. 168–181.

[12] EL FAOUZI N.E. Nonparametric traffic flow prediction using kernel estimator. In: *Transportation And Traffic Theory. Proceedings Of The 13th International Symposium On Transportation And Traffic Theory, Lyon, France, 24-26 July 1996*, 1996.

[13] ELMAN J.L. Finding structure in time. *Cognitive science*, 1990, 14(2), pp. 179–211.

[14] GRAVES A.B. System and method for speech recognition using deep recurrent neural networks, February 5 2019. US Patent App. 15/043,341.

[15] HOCHREITER S., SCHMIDHUBER J. Long short-term memory. *Neural computation*, 1997, 9(8), pp. 1735–1780.

[16] JAFARPISHEH N., TESHNEHLAB M. Cancers classification based on deep neural networks and emotional learning approach. *IET systems biology*, 2018, 12(6), pp. 258–263.

[17] JEONG Y.S., BYON Y.J., CASTRO-NETO M.M., EASA S.M. Supervised weighting-online learning algorithm for short-term traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, 2013, 14(4), pp. 1700–1707.

[18] KAMARIANAKIS Y., PRASTACOS P. Forecasting traffic flow conditions in an urban network: Comparison of multivariate and univariate approaches. *Transportation Research Record: Journal of the Transportation Research Board*, 2003, 1(1857), pp. 74–84.

[19] KUMAR K., PARIDA M., KATIYAR V.K. Short term traffic flow prediction in heterogeneous condition using artificial neural network. *Transport*, 2015, 30(4), pp. 397–405.

[20] LEVIN M., TSAO Y.D. On forecasting freeway occupancies and volumes (abridgment). *Transportation Research Record*, (773), 1980.

[21] LEWANDOWSKI M., PŁACZEK B., BERNAS M., SZYMAŁA P. Road traffic monitoring system based on mobile devices and bluetooth low energy beacons. *Wireless Communications and Mobile Computing*, 2018.

[22] LI Y., LI Z., LI L. Missing traffic data: comparison of imputation methods. *IET Intelligent Transport Systems*, Feb 2014, 8(1), pp. 51–57.

[23] LV Y., DUAN Y., KANG W., LI Z., WANG F. Traffic flow prediction with big data: A deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, April 2015, 16(2), pp. 865–873.

[24] MIKOLOV T., KARAFIÁT M., BURGET L., ČERNOCKỲ J., KHUDANPUR S. Recurrent neural network based language model. In: *Eleventh annual conference of the international speech communication association*, 2010.

[25] NIHAN N.L. Aid to determining freeway metering rates and detecting loop errors. *Journal of Transportation Engineering*, 1997, 123(6), pp. 454–458.

[26] SUN S., ZHANG C., YU G. A bayesian network approach to traffic flow forecasting. *IEEE Transactions on intelligent transportation systems*, 2006, 7(1), pp. 124–132.

[27] TAYLOR M.A.P., BONSALL P.W. *Understanding traffic systems: data analysis and presentation.* Routledge, 2017.

[28] WILLIAMS B. Multivariate vehicular traffic flow prediction: evaluation of arimax modeling. *Transportation Research Record: Journal of the Transportation Research Board*, 2001, (1776), pp. 194–200.

[29] WILLIAMS B.M., HOEL L.A. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of transportation engineering*, 2003, 129(6), pp. 664–672.

[30] XU Q.H., YANG R. Traffic flow prediction using support vector machine based method. *Journal of Highway and Transportation Research and Development*, 2005, 22(12), pp. 131–134.

[31] ZHENG W., LEE D.H., SHI Q. Short-term freeway traffic flow prediction: Bayesian combined neural network approach. *Journal of transportation engineering*, 2006, 132(2), pp. 114–121.

[32] ZREIK M., VAN HAMERSVELT R.W., WOLTERINK J.M., LEINER T., VIERGEVER M.A., ISGUM I. Automatic detection and characterization of coronary artery plaque and stenosis using a recurrent convolutional neural network in coronary CT angiography. *arXiv preprint arXiv:1804.04360*, 2018.

[33] POLSON N.G., SOKOLOV V.O. Deep learning for short-term traffic flow prediction. *Transportation Research Part C: Emerging Technologies*, 2017, 79, pp. 1–17.

[34] KOESDWIADY A., SOUA R., KARRAY F. Improving traffic flow prediction with weather information in connected cars: A deep learning approach. *IEEE Transactions on Vehicular Technology*, 2016, 65(12), pp. 9508–9517.

[35] WU Y., TAN H., QIN L., RAN B., JIANG Z. A hybrid deep learning based traffic flow prediction method and its understanding. *Transportation Research Part C: Emerging Technologies*, 2018, 90, pp. 166–180.

[36] FU R., ZHANG Z., LI L. Using LSTM and GRU neural network methods for traffic flow prediction. *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, 2016, pp. 324–328.

[37] ZHENG H., LIN F., FENG X., CHEN Y. A hybrid deep learning model with attention-based conv-lstm networks for short-term traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[38] ZHANG W., YU Y., QI Y., SHU F., WANG Y. Short-term traffic flow prediction based on spatio-temporal analysis and CNN deep learning. *IEEE Transactions on Intelligent Transportation Systems*, 2019, 15(2), pp. 1688–1711.

[39] ZHANG X., GAO Y., LIN J., LU C.-T. TapNet: Multivariate Time Series Classification with Attentional Prototypical Network. *AAAI*, 2020, pp. 6845–6852.

[40] SHOKOOHI-YEKTA M., WANG J., KEOGHR E. On the non-trivial generalization of dynamic time warping to the multi-dimensional case. *Proceedings of the 2015 SIAM international conference on data mining*, 2015, pp. 289–297.

[41] KARIM F., MAJUMDAR S., DARABI H., HARFORD S. Multivariate LSTM-FCNs for time series classification. *Neural Networks*, 2019, 16, pp. 237–245.