DRGNN – DILATED RECURRENT GRAPH NEURAL NETWORK FRAMEWORK INCORPORATING SPATIAL AND TEMPORAL FEATURES SIGNIFYING SOCIAL RELATIONSHIPS IN IOT NETWORK BASED TRAFFIC PREDICTION

J. Divya[*], *A. Chandrasekar*[†]

Abstract: The intelligent transportation system seeks to reduce traffic and improve the driving experience. They give us a lot of data that we can use to improve services for both the public and transportation officials by feeding it into machine learning systems. Most importantly, Traffic environment refers to everything that might have an impact on how much traffic is moving down the road, including traffic signals, accidents, protests, and even road repairs that might result in a backup. A motorist or rider can make an informed choice if they have previous knowledge that is very close to approximate all the above and many more real-world circumstances that can affect traffic. Additionally, it aids in the development of driverless vehicles. Traffic data have been growing dramatically in recent decades, and we are moving toward big data concepts for transportation. The current approaches for predicting traffic flow use some traffic prediction models, however they are still inadequate to handle practical situations. We thus aimed to focus on the traffic flow forecast problem using the traffic data and prediction models. The proposed model called DRGNN, a dilated recurrent graph neural network framework aims to effectively analyze and predict the traffic pattern by considering the spatial (space) and temporal (time) aspects of the real-time traffic data considering social relationships between internet of vehicles which indeed produced accurate and valuable insights that could help in deploying the model in any suitable real-time traffic monitoring and prediction system.

Key words: *graph neural networks, social internet of things, traffic prediction*

Received: ? **DOI:** 10.14311/NNW.2023.33.026
Revised and accepted: ?

[*]Jegatheesan Divya – Corresponding author; Information Technology, St. Joseph's College of Engineering, Chennai, India, E-mail: divyaj@stjosephs.ac.in

[†]Arumugam Chandrasekar; Computer Science and Engineering, St. Joseph's College of Engineering, Chennai, India

1. Introduction

The internet of things (IoT) refers to a new world in which billions of smart objects automatically communicate and interact with one another. A new paradigm—the social internet of things (SIoT) in which IoT and social networks are combined in a novel way pertaining to be one of the most important areas in this sector. A complete review of the SIoT system [1] has been analyzed and the social IoT based existing scenario in smart cities was explored. SIoT could be elucidated using a broader sense that primarily comprises of six components namely relationship management, architecture, trust management, information, web services, and SIoT tools, which includes datasets and platforms. Intelligent Traffic Management Systems (ITMS) were established as a result of the rise of the internet of things and its use in smart cities, which provides the ideal platform for tackling traffic-related challenges. In terms of determining the best route, lowering average wait times, traffic congestion, the cost of travel, and the level of air pollution, our proposed approach aids in resolving the multiple difficulties that traffic management authorities confront. Graph neural networks (GNNs), an innovative and rapidly expanding lineage of neural network models, can interpret multifarious communications happening. in the object layer and have been demonstrated to produce cutting-edge results across various challenges in IoT environments. A survey where GNN [2] can be employed in IoT networks has been investigated. Graphs in a variety of real-world contexts [3], including transportation networks, social networks, and skeleton-based human behaviors, include both spatial and time fluctuations. With variable node distributions and different edge linking relations, the topological structures of graphs change over time. The goal of the proposed DRGNNs is to simulate the spatial and temporal dependencies in dynamic graphs integrating the concept of social internet.

2. Related works

In this section, various GNN approaches emphasized in the literature have been reviewed first, followed by the role of GNN in traffic prediction, and next the impact of SIoT in various real time applications is discussed.

Zhao et al. [4] proposed a novel network architecture called persistence enhanced GNN which influenced permanent homological information to reassign weights to the messages sent between the nodes in the graph during the time of convolution which helps to utilize the structural information in real time graphs. It involved fewer computational steps with back propagation learning, but attention was required in order to increase the accuracy rate. Wang et al. [5] discussed a GNN based approach for predicting protein B-factor in a biophysical breakdown by constructing persistent spectral graphs to expose both topological persistence and geometric shape from high-dimensional datasets which proved a notable performance in molecular data, but it did not integrate any machine learning models to improvise the prediction accuracy. Casas et al. [6] proposed a new methodology called SPAGNN, spatially aware graph neural networks to produce a socially consistent, probabilistic estimation of future trajectories of autonomous vehicles based on object detection and relational behaviour forecasting. Raster maps with various information about roads, traffic lights, and lanes were encoded in distinct channels in order to facilitate convolutional neural network (CNN) learning. However, agents like cycles and pedestrians were not considered in trajectory prediction. Xie et al. [7] suggested a graph neural based algorithm for predicting traffic speed prediction in urban roads and named a sequential graph neural network (SeqGNN). In this paper, the authors have constructed a graph model using the connectivity of road segments in which the properties of road segments are mapped as nodes, and the connections between them are represented as edges. But the external influential factors like weather and points of interest were not considered. Peng et al. [8] Proposed a prediction technique called DynGRCNN, dynamic graph recurrent convolutional neural network meant for estimating urban traffic passenger flow which integrated the relationship among passengers with a GNN and added spatial temporal features in the learning mechanism. It was able to achieve better accuracy but was not applicable to complex traffic order forecasting and route planning tasks were not accomplished. Diehl et al. [9] suggested that including plenteous interactions in the traffic dataset could decrease the prediction error by 30% and increase accuracy and so proposed the GNN for modelling traffic participant interaction. The authors also concluded that integration of this model with recurrent neural network could be tried out. Gao et al. [10] proposed a knowledge based graph convolutional network by introducing supplementary info as position vectors integrating structure and text representations using a gate mechanism to perform classification and prediction tasks. This model has not considered category information to classify the entities.

In 2010, the first significant step in fusing social networks with IoT was taken by Kranz et al. [11], wherein the authors suggested that people might use their own smart objects as well as those of their friends to share the services they provide. Hence, social networks and IoT are successfully integrated to establish an effective relationship. Subgraph matching with dynamic weight (SMDW), a subgraph matching alternative has been proposed by Jiang et al. [12] which was able to overcome the issue of the core subgraph getting updated whenever the changing communities based on user's interest were tracked. The tests built on real datasets have been created to assess the performance of the suggested model by contrasting it with leading traditional models in this field and the data processing has been completed through the edge layer. Wang et al. [13] suggested an intercommunity detection method based on compressive sensing (CS) over graphs. The measurement matrix is constructed by using the likelihood of two nodes encountering one other, and the encounter probability and edge clustering coefficient are utilized to build a new metric for each connection. The intercommunity linkages are then detected using a CS-based detection technique which further requires real-time management and analysis. Perera et al. [14] analyzed the primary concept of enabling sensing of IoT data as a service model architecture, by examining the socio-economic benefits and challenges associated with the socio-IoT community. Therefore, providing sensing as a service, location-based recommendations also play an important role. Hu et al. [15] Proposed a graph based IoT model meant for service recommendation and automation where context-aware data was taken to construct the IoT context graph with different semantic relationships and effective biased random walk method was followed to capture the neighborhoods of nodes. Heterogeneity was achieved where the complexity was high. Friji et al. [16] devised a framework to aid multi-user coordinated navigation to promote safety for pedestrians during COVID. Weighted graphs representing the social relations connecting the different IoT devices were constructed, and community detection was performed based on the social relations along with graph based routing was carried out but was not scalable.

From the above literature works and studies carried out, we could infer that the potential of graph application in social network of things is broad, extensive, and still requires a comprehensive structural analysis and augmented graph systems to populate precise, stable real-time implementation scenarios. Therefore, we propose DRGNN model that primarily focuses on serving as an ideal platform for tackling traffic-related challenges in terms of determining the best route, lowering average wait times, traffic congestion, the cost of travel, and others among the socially extended connections.

Section 3 discusses the modelling and implementation of DRGNN with detailed algorithms, Section 4 analyses the results obtained, and Section 5 describes the conclusion and future work respectively.

3. Methodology

A conventional way of using graphs in convolution neural networks [17] was suggested which in turn lead to the inception of graph convolutions. Graph convolution networks (GCNs) are a type of neural network that may be used to learn from graphs. GCN's basic concept is to apply convolution on a graph. GCN takes a graph as input instead of a two-dimensional array in a traditional convolutional network.

The purpose of graph convolution [18] enables to determine signal/feature function on a graph $G = (V, E)$. We present an efficient DRGNN – dilated recurrent graph neural network model – a graph based recurrent network model as shown in Fig. 1 that incorporates spatial and temporal features captured simultaneously from traffic data on a dilated graph network integrated with GRU (gated recurrent unit). The algorithm involves the collection of data, construction of adjacency matrix, achieving dynamicity via the constructed and refined complex graph network, friendship selection, and obtaining the feature embeddings via the graph convolutions and forecasting the outputs.

3.1 Dataset preparation and graph construction

A collection of vehicle traffic datasets collected over six months between two sites during a given length of time with total observation points of count 449. The road network is labeled as G from the raw (CSV) and city pulse information model with required metadata. Initially, the topological structure of the road network is described by an unweighted graph $G = (V, E)$, where each road is regarded as a node, with $V = v_1, v_2, \dots, v_n$, the number of nodes is denoted by n and E being a collection of edges. To express the link between roadways, the adjacency matrix $\mathbf{A}$ is utilized. We view the traffic data on the road network G as an attribute feature of each node in the network. Any traffic related information, namely density,

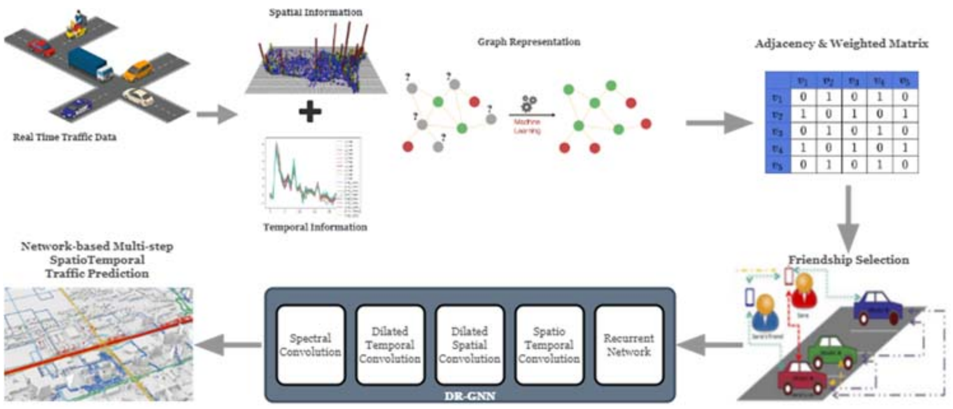
Real Time Traffic Data
Spatial Information
Temporal Information
Graph Representation
Adjacency & Weighted Matrix
Friendship Selection
Network-based Multi-step SpatioTemporal Traffic Prediction
Spectral Convolution
Dilated Temporal Convolution
Dilated Spatial Convolution
Spatio Temporal Convolution
Recurrent Network
DR-GNN

Fig. 1 *DRGNN architecture.*

speed, and flow of traffic can be included in the node attribute features. Based on the topology of road network G and attribute features the calculation of traffic information in the next t seconds can be computed.

3.2 Obtaining spatial and temporal dependence

In traffic forecasting, acquiring the complicated spatial dependence [19] is a major challenge. The CNN model can capture the intricate topology of an urban road network since it is shaped more like a graph than a two-dimensional grid and thus captures spatial dependence effectively. Using GCN model, the topological link between the center road and its surrounding roads can be calculated encapsulating road attributes and the topological structure of the road network and then the spatial dependence can be computed. Temporal aspects of action segmentation in videos and detection of frames were analyzed in the literature [20] initially. According to our suggested methodology, the network receives the current traffic data as well as the concealed status at time $t-1$ to determine the traffic state at time t. While the model captures current traffic information, it also retains the changing pattern of prior traffic data and possess potential to effectively acquire the temporal dependence.

3.3 Link and relations between devices

SIoT devices have a variety of social relationships. The information about the devices, such as who owns them and where they are located geographically helps to create connections between the devices. The following three social ties [21] are considered in this methodology based on the relations between entities:

(i) *Co-location based relation (CLOR):* The spatial characteristics of the devices are used to infer this relationship. As a result, if a group of devices exists in a specific location, CLOR relationships exist between them. The devices can either be stationary or move around. As a result, during roaming their CLOR relationships with other devices can be modified on the fly.

(ii) *Social object relation (SOR):* When two gadgets interact in a constant or sporadic manner, a SOR is formed. The requirements for establishing the relationships are dictated by the policies of the owners. A SOR relationship can be established between two devices if they are co-located and communicate data for a set amount of time. For example, between two smart cars when they are adjacent together in traffic.

(iii) *Social friendship relation (SFR):* High-weight linkages are established between devices that have the same owner that is from the same source in case of vehicles. The owner's social network can then be utilized to create less weighted relationships between devices which relies on how many friends each owner can have either as immediate friends or friends of friends, and then gets devised onto the SIoT network.

3.4 Algorithm construction and implementation

The implementation of the proposed DRGNN methodology is depicted sequentially in the algorithms defined hence forth. Algorithm 1 focuses on constructing a graph network topology from the numerical road data (primarily based on latitude and longitude) and establishing relationships between each of the nodes. A weighted adjacency graph matrix is obtained. The weighted matrix is optimized and refined to obtain the closest neighbors in Algorithm 2. Algorithm 3 mainly focuses on friendship selection between the selected nodes to pass on to the graph convolutions. Algorithm 4 obtains the spectral, temporal, and spatio-temporal aspects of the nodes upon which the hyper-parameters are fined tuned, neural net layers are modified and model inference is established. Spatial and temporal parameters will be the dependent variables of our model based on which traffic pattern is predicted.

3.4.1 Algorithm 1 – Construction of adjacency matrix

The traffic data is converted into a complex network where the relationships between vehicles are established. A vehicle would build a relationship with N vehicles (note that vehicles are considered as nodes) in that defined range of distance and at that time instant. Thus, the spectral (distance and position of vehicle) and the temporal at a time instants $T, T + k, T + 2k$ (for instance 5, 10, 15 seconds) aspects of the traffic data are obtained and analyzed in the graph based convolutional networks.

Edge calculation and establishment of relationship is done in the following ways:

1) If any two vehicles (vertices) are in the same location (same latitude and longitude) and assume that the distance between the vehicles is less than 50 meters, establish a SFR (social friendship relation) between the vertices. Note that the distance between the vehicle can be varied based on the application.

2) If any two vehicles, already in the SFR relationship, have a sideway connection (i.e., mathematically in the same row and distance is + or -50 meters), then establish a CLOR (co-location object relationship) between the vehicles.

3) If there exists a main start point and all the vehicles (vertices) project themselves from that point, establish a POR (parent object relationship)

4) The *edgeWeight* values are computed to construct the adjacency matrix $\mathbf{A}\{V, E\}$. A non-zero entry at position (i, j) indicates a connection (edge) between vehicles i and j. The value in the entry (i, j) represents the *edgeWeight* between vehicles i and j.

Algorithm 1 Construction of adjacency matrix.

Input data: Nodes with vertices $V\{v_1, v_2, \ldots, v_n\}$, number of nodes n, edges $E\{e_1, e_2, \ldots, e_m\}$, number of edges m, graph $G(V, E)$
Output data: Adjacency matrix $\mathbf{A}\{V, E\}$ comprising of all the nodes (vehicles) and their routes (edges) with V vertices and E edges in $(V \times E)$ dimension

```
// $i$ ranges from 1 to $n$
for $\forall v_i$ do
  // $j$ ranges from 1 to $m$
  for $\forall e_i$ do
    if $sameRoute(v_i, e_j) == 1$ then
      if $sameLocation(v_i, e_j) == 1$ then
        if $distance(v_i, e_j) <= 50$ meters then
          // establishSFRrelation()
          if $sidewaysExists(v_i, e_j) == 1$ then
            // establishCLORrelation()
            if $centralStartPoint(v_i, e_j) == 1$ then
              // establishPORrelation()
              $edgeWeight = 1$
            else
              $edgeWeight = distance(v_i, e_j)$
            end if
            $establishCLORrelation(v_i, e_j, edgeWeight)$
          end if
          $establishSFRrelation(v_i, e_j, edgeWeight)$
        end if
        $establishFriendship(v_i, e_j, edgeWeight)$
      end if
    end if
    $\mathbf{A}\{V, E\} \leftarrow edgeWeight$
  end for
end for
```

3.4.2 Algorithm 2 – Establishing weighted matrix

To obtain an optimized and refined weighted matrix $\mathbf{M}_w$ by finding closest neighbours and removing nodes with negligible weights. Each time the network graph changes, embeddings will also be updated to achieve dynamicity, Algorithm 2 is proposed as follows:

Algorithm 2 Establishing weighted matrix.

```
Input data: Adjacency matrix – $\mathbf{A}\{V, E\}$ with $V \times E$ dimension with $edgeWeight$
Output data: Weighted matrix $\mathbf{M}_w$ comprising of selective neighbour nodes

// compute node embedding $R_i$
define $R_i$
//$deg(v_i)$ represents the number of edges connected to $v_i$
$R_i \to deg(v_i) \times (\sum_{i \in neighbours} V(i,j))$
// define temporary node
$\eta_t \to tempNode()$
$S_{\text{start}} \to R_i / \sum_i R_i$
for $i$ : 1 to $n$ do
  $V_{(i,-)} \leftarrow selectNode(S_{\text{start}})$
  $defineStart(\eta_t) \to V_{(i,-)}$
  //dynamicity where location/position/columns vary
  for $j$ : 1 to $m$ do
    if $neighbourExists(V_{i,j}) \neq edgeWeight$ then
      $V_{(i+1,j+1)} \leftarrow selectClosestNeighbour()$
    else
      $V_{(i+1,j+1)} \leftarrow selectPreviousVertex()$
    end if
    $removeLinks()$
    $updateWeights()$
  end for
  include $\eta_t$ into matrix $\mathbf{M}_w$
end for
```

3.4.3 Algorithm 3 – Friendship selection

Friendship selection – a mechanism in which every object can use its friendships to find the needed service in a distributed fashion, using only local knowledge. Multiple efficient friendship selection mechanisms for social IoT27 were indeed observed that this paper suggested the use of an external object called a smart social agent that is required to establish a relation between the objects or nodes. Thus, the selection of social agents within the social IoT network could be quite challenging. The parental object relationship (POR) is described as a relationship between similar products or vehicles created by the same manufacturer over the same time period. Furthermore, objects, like humans, can form co-location object relationships (CLOR) and co-work object relationships (CWOR) when they share their private like cohabitation or public and professional working experiences. Another sort of relationship is the ownership object relationship (OOR), which is specified for vehicles, smart phones etc. owned by the same user. The social object relationship (SOR) is created when two or more objects come into contact for only social reasons (e.g., when friends' devices or sensors come into contact). The friendship selection process depends on the parameters established by these relationships.

Algorithm 3 Friendship selection.

```
Input data: Vertices, $v_i$ and $v_{i+1}$ from the weighted matrix $\mathbf{M}_w$ comprising of
selective neighbour nodes
Output data: Selective road topology graph with established relationships

Consider the vertices, $v_i$ and $v_{i+1}$
if $weight(v_i) == weight(v_{i+1})$ then
  addFriend()
else
  findFriend()
  check weight and type of friendship
  if $(isExists(friend(v_{i+1})))$ then
    $graph(type, v_i, weight)$
  else
    graph(dummyNode)
  end if
end if
```

3.4.4 Algorithm 4 – Construction of DRGNN

Obtaining the spectral, temporal, and spatio-temporal aspects of the nodes in order to construct the DRGNN model is depicted in Algorithm 4 as follows:

Algorithm 4 Construction of DRGNN.

```
//Define parameters – param()
Let $learningRate = 0.001$, $epochs = 50$, $batchSize = 32$ & $64$,
    $trainRate = 0.8$, $sequenceLength = 5$

//Define layers

a) SpectralConvolution()
Input: tensor input $\mathbf{X}$, trainable parameter $\theta$, represented as diagonal matrix
called as parameterized filter in spectral domain input channel size $s_{\text{in}}$,
output channel size $s_{\text{out}}$, kernel input size $k$
Output: spectral tensor output $\mathbf{mat}_{\text{sconv}}$, kernel output size $\eta$, $\mathbf{Y} \leftarrow \mathbf{X}[s_{in} \times k][\eta]$,
where $\mathbf{Y}$ is the reshaped version of tensor input $\mathbf{X}$.

// Note that it is important to reshape all the calculated values to the output
kernel size
// Apply transpose to $\mathbf{X}$, reshaping the dimensions
    $\mathbf{X} \leftarrow reshape([\mathbf{X}]^{\mathrm{T}}, [-1, \eta])$
    // kernel matrix will be in dimension $k \times k$
    // Transform the node features (graph signal) into the spectral domain
    $\mathbf{temp}_{\text{mat}} = \mathbf{X} \cdot \mathbf{kernel}$
    // reshaping the dimensions
    $\mathbf{temp}_{\text{mat}} \leftarrow reshape(\mathbf{temp}_{\text{mat}}, [-1, s_{\text{in}}, k, \eta])$
    // Apply transpose to $\mathbf{temp}_{\text{mat}}$
    $\mathbf{mat}_{\text{ker}} \leftarrow reshape([\mathbf{temp}_{\text{mat}}]^{\text{T}}, [-1, s_{\text{in}}, k])$
    // filter application
    $\mathbf{mat}_{\text{sconv}} \leftarrow ([\mathbf{mat}_{\text{ker}}] \times \theta])\cdot\ reshape(\eta, s_{\text{out}})$
    // spectral tensor output
    $return(\mathbf{mat}_{\text{sconv}})$

b) $TemporalConvolution()$
    **Input:** tensor input $\mathbf{X}$, kernel size $k$, time $t$,
    $\mathbf{X}_{\text{in}} \leftarrow \mathbf{X}[:, k-1 : t, :]$, dilation stride $r$
    **Output:** convolved weights $\mathbf{WT}$ with respect to time $t$ where $t \in \{1, \ldots, t_{k-1}\}$

    // Steps to perform temporal convolution with dilation and apply a GLU
    activation
    define  $GLU(\mathbf{X})$
    // $GLU = linear + sigmoid$
    Linear function: $\mathbf{l}(x) = \mathbf{m}(x) + b$
    Sigmoid function: $\mathbf{s}(x) = \frac{1}{1+e^{-x}}$
    // dimensions $[k, s_{\text{in}}, 2s_{\text{out}}]$ define the shape of the weight matrix,
    // where each element in the matrix represents a weight parameter
    // to be learned by the model
    $\mathbf{WT} = \mathbf{matrix}_{[k, s_{\text{in}}, 2s_{\text{out}}]}$
    $bias\_time\ = \mathbf{matrix}[0]_{[2s_{\text{out}}]}$
    // $bias\_time$  is a constant added to the weights to maintain stability
    $X_{\text{conv}} = (\mathbf{X}_{\text{in}}, \mathbf{WT}) +\ bias\_time$
    // $k_t$ represents kernel size in temporal dimension
    $X_{\text{conv}} = dilations(filterEnlarge(k_t + (k_t - 1)(r - 1)))$
    $return(X_{\text{conv}}[0 : s_{\text{out}}] + GLU(\mathbf{X}_{\text{in}}))$

c) $SpatialConvolution()$
**Input:** Input channel size $s_{\text{in}}$ and weighted matrix with spectral feature vectors
$\{$Lat, Long$\}$, dilation stride $r$
**Output:** Convolved weights $\mathbf{WS}$ with respect to spatial factors $S \in \{S_{\text{Lat}_{(i)}} S_{\text{Long}_{(i)}},$
$\ldots, i+1\}$

    // Steps to perform spatial convolution with dilation
    $\mathbf{WS} = \mathbf{matrix}_{[k_{\text{s}}, s_{\text{in}}, s_{\text{out}}]}$
    $bias\_space\ = \mathbf{matrix}[0]_{[s_{\text{out}}]}$
    $X_{\text{spatial}} = spectralConvolution(\mathbf{X},\ [-1, \eta, s_{\text{in}}], k_{\text{s}}, s_{\text{in}}, s_{\text{out}}\ ) +\ bias\_space$
    $X_{\text{spatial}} = dilations(filterEnlarge(k_{\text{s}} + (k_{\text{s}} - 1)(r - 1)))$
    $X_{spatialOut} = reshape(X_{\text{spatial}}, (\eta, s_{\text{out}}))$
    $return(X_{\text{spatialOut}} + X_{\text{input}})$

d) $spatio\text{-}temporalConvolution()$
    // where $c_t$ represents input channel size and $c_{t-1}$ represents output channel
    // size from the previous time step ($t$–1)
    // where $c_s$ represents input channel size and $c_{t+1}$ represents output channel
    // size from the next time step $(t+1)$
    $layer = temporalConvolution(\mathbf{X}, k_t, c_s, c_{t-1})$
    $layer = spatialConvolution(\mathbf{X}, k_t, c_t, c_{t+1})$
    $X_{\text{final}} = layerNormalization(layer)$
    $return(X_{\text{final}})$

e) $recurrentNetwork()$
Input: current state $h_t$ , previous state $h_{t-1}$ and input state $X_t$
Output: output layer $Y_t$, weight at output $W_{\text{hy}}$

    // update of the current state $h_t$ based on the previous state $h_{t-1}$ and the
    // input state $X_t$
    $h_t = f(h_{t-1}, X_t)$
    // specific calculation for updating $h_t$, where $W_{hh}$ is the weight for the
    // recurrent connection and $W_{xh}$ is the weight for the input connection
    $h_t = \tanh(W_{\text{hh}} h_{t-1} + W_{\text{xh}} X_t)$
    // the output $Y_t$ is obtained by multiplying the current state $h_t$
    // by the weight $W_{\text{hy}}$
    $Y_t = W_{\text{hy}} h_t$
    $\rightarrow$ update through $X_1, \ldots, X_{n-1}$ layers recurrently

f) $mainFunction()$
    $graph(weight\ (W),\ vertices\ V(i,j))$

g) $preprocess()$
    $scaledLaplace(W)$
    // where $d[i]d[j]$ represents diagonal elements
    $\mathbf{L}[i,j] = \frac{L[i,j]}{\sqrt{d[i]d[j]}}$
    // where $\lambda$ represents regularization parameter, $\mathbf{identity}_{\text{mat}}$ represents
    // identity matrix
    $return(2\mathbf{L}/(\lambda - \mathbf{identity}_{\text{mat}}))$

h) $normalize()$
    // Weights are normalized within the range of 0 to 1
        $\mathbf{W} \leftarrow \mathbf{W}[0,1]$

i) $train()$
    // train the model
    $buildLayers(\mathbf{X}, k, s_{\text{in}}, s_{\text{out}})$

j) $defineParameters()$
    $modelInference(pred,\ inputs,\ batchsize,\ step\_index,\ step\_time)$

k) $saveModel()$
```

This algorithm primarily focuses on obtaining the temporal and spatial aspects of the graph data to bring in the traffic flow forecast and recommendation. The initial parameters such as epochs, batch size, learning rate are defined. The spectral convolution layer focuses on obtaining the trainable parameters and outputs the spectral tensor with a given kernel size and weights. The temporal convolution obtains the time aspect also as input parameter, and outputs the updated weights. The dilation adds kernel level skipping to the layers that yields to even more precise prediction. The spatio-temporal convolution combines the weighted outputs of space and time aspect with regard to traffic data and the updated weights are normalized to obtain the final output. The recurrent network creates a cycle for the sequence of traffic data and makes predictions. The kernel size is obtained, and the output is normalized. Thus, the model is saved and trained with the inputs, and inference is made.

4. Results and discussion

The traffic forecast using DRGNN are employed, and the observations made are discussed. The primary focus of the graph-based convolution is employed on the real time traffic data by the proposed methodology. The DRGNN model is built and evaluated upon the traffic data that serves the purpose of forecasting the traffic pattern and interpreting the results to the user.

4.1 Insights of dataset

We use the historical obtained from the dataset as input and compute the final prediction through the graph convolutional neural network. The dataset is accessible under the name "CityPulse Smart City Datasets" on the internet which includes various attributes like vehicle id, latitude, longitude, timestamp, distance in meters, type of road, etc. Totally 449 observation points were derived from a collection of vehicle traffic datasets collected over a six-month period between two sites during a given length of time. A combination of four datasets is obtained. The duration of traffic captured includes a period of data captured with a time frame difference of six months, say, from February to July. An adjacency matrix and a feature matrix are generated from the data. The adjacency matrix in traffic networks is computed based on the separation between sensors in terms of distance. The dataset contained some missing data, therefore we used the linear interpolation approach to fill in the missing values. In the experiments, the input data was standardized to the range $[0, 1]$. In addition, a training set of 80% of the data and a testing set of 20% of the data were used in this study.

4.2 Model setup

Our model's key parameters are batch size, the number of hidden layers, training epoch, and learning rate (LR). In the experiment, we manually changed the learning rate to 0.001, the batch size to 64, and the training epoch to the maximum of 150. As different hidden units can significantly affect prediction accuracy, the number of hidden units in the model is an important parameter. In order to get the optimal value, we tested with several hidden units before comparing the predictions made for various epochs.

4.3 Evaluation metrics

The following listed metrics are taken into consideration for the parametric evaluation of our model namely root mean square error (RMSE), mean absolute error (MAE), R^2, accuracy (acc) and variance (var) as per the Eqs. (1), (2), (3), (4) and (5)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_l\right)}, \tag{1}$$

$$MAE = \frac{1}{n}\sum\nolimits_{i=1}^{n}\left|Y_i - \hat{Y}_l\right| \tag{2}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(Y_i - \hat{Y}_l\right)^2}{\sum_{i=1}^{n}\left(Y_i - \bar{Y}\right)^2}, \tag{3}$$

$$acc = 1 - \frac{\left\|Y - \hat{Y}\right\|_F}{\|Y\|_F}, \tag{4}$$

$$var = \frac{\sum\limits_{i=1}^{k}\left(Y_i - \hat{Y}\right)^2}{k-1}, \tag{5}$$

where Y_i represents the actual values and $\hat{Y}_l$ denotes the predicted values of Y at time l, n is the number of observations or data points, Y_i represents each individual data point, $\bar{Y}$ is the mean of the actual values and k represents the number of folds.

4.4 Result analysis

The DRGNN model that operates on the traffic data is evaluated and the results are also obtained for the weather analysis and pollution analysis. The Tab. I represents the different parameters based on which the traffic data is analyzed. The minimum

LR	Batch size	Epochs	acc	RMSE	MAE	R^2	var
0.0001	32	50	0.5795	14.5698	14.2340	0.8532	0.8631
	64	75	0.6923	12.0321	11.3450	0.6231	0.6012
	64	100	0.8143	10.9030	7.5483	0.3820	0.3874
0.00001	64	25	0.7743	11.6701	10.4950	0.5784	0.5892
	64	50	0.8212	10.5001	7.3049	0.4269	0.4890
	64	150	0.8991	6.5272	4.3867	0.7785	0.7721

Tab. I *Hyper-parameter performance of DRGNN model.*

RMSE value, minimum MAE, the maximum accuracy, variance factors of the model that is trained for different parameters are listed.

Tab. I shows the performance of the proposed model that is trained for different learning rates. It is evident that the model yielded a maximum accuracy of 89%, with a minimum RMSE value of 6.52, MAE score of 4.38, variance of 0.77, with a batch size of 64 trained for 150 epochs and learning rate of 10^{-5}. The model, when trained for a batch size of 64 for 100 epochs and learning rate of 10^4 produced an output of maximum accuracy 81% and minimum RMSE value of 10.90, minimum MAE score of 7.54, R^2 value of 0.38.

The parameter output of the actual and predicted result depicts the near and far ranges in which the predicted pattern differs from the actual forecast for the model trained for 75 epochs and 100 epochs thereby as shown in Fig. 2 and 3 when the model is trained with a learning rate of 10^{-4}. Fig. 4 shows the graphical representation of the performance of the model when trained with a learning rate of 10^{-5}.

The RMSE values during training initially appears to be discrete when trained for minimal epochs. Later on, which the RMSE values are found to be continuous mainly due to varying dynamicity of the nodes and improved gradient factors. The loss is also observed to reduce with steep curve and attain a fixed threshold. The MAE score is found to jump between minimal variant values and reach a stable point. An accuracy of 82% and 89% was achieved for the model when trained for 50 and 150 epochs respectively for the given spatial (space – location, latitude, longitude) and temporal aspect with respect to time frame. The model was trained

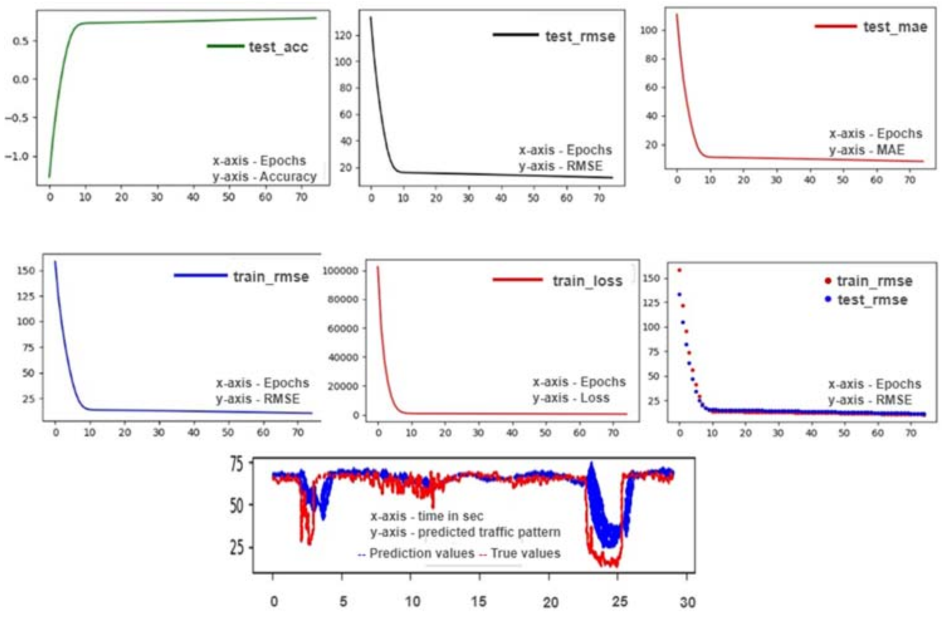

Fig. 2 *DRGNN model trained for LR 10^{-4} for 75 epochs.*

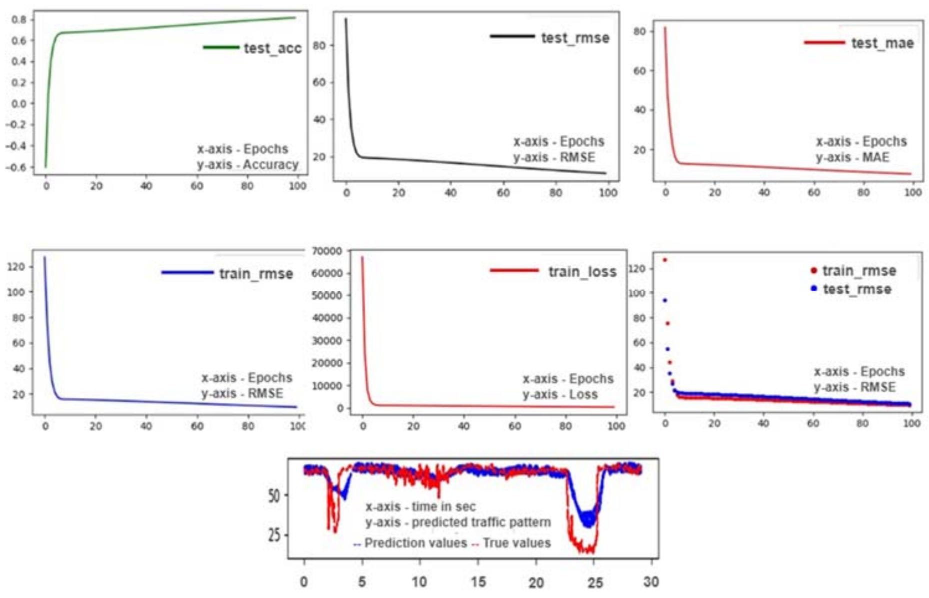

Fig. 3 *DRGNN model trained for LR 10^{-4} for 100 epochs.*

with better optimization function called Adam optimizer and dilations were added, which is chosen to be the alternated option for the enhanced model so as it suits well for sparse data with dynamic learning rate which led to a maximum accuracy of 89% with dynamic nodes for 150 epochs. The dynamicity of graph nodes indeed was achieved using a random function that led to the constructive selection of different node every time the model is re-run. The prediction traffic pattern was forecasted for a time space of 5, 10 and 15 seconds with the spatial aspect of varying latitude and longitude. The dynamicity factor was achieved as every time the model is run, the nodes are allocated randomly, and the topology is reconstructed dynamically.

Tab. II shows the constructive comparison between different models that are evaluated for the given scenario. The prediction results of the proposed model seem to elevate better when compared with other conventional models say density-based spatial clustering of applications with noise (DBSCAN), historical average (HA) and auto regressive integrated moving average (ARIMA). For a time instant T, the accuracy and RMSE of HA, ARIMA are 87%, 7.44 and 82%, 10.04 respectively where the DRGNN seems to perform well with a precision accuracy of 85% and RMSE score of 5.89. For any arbitrary additive value of k, the DRGNN model outperforms with a maximum accuracy of 89% ultimately. The GCN and DRGNN model shows slight differences with the performance where the MAE, Accuracy, and variance are significantly better for the proposed model with 4.123, 87%, 0.732 values for time instant $T + k$. The loss values of the trained model seem to be negligible. From the results as tabulated and depicted above, we could conclude that the implication of graph based network in traffic prediction, finds the best application as different parameters could be analyzed at a single stretch with minimal

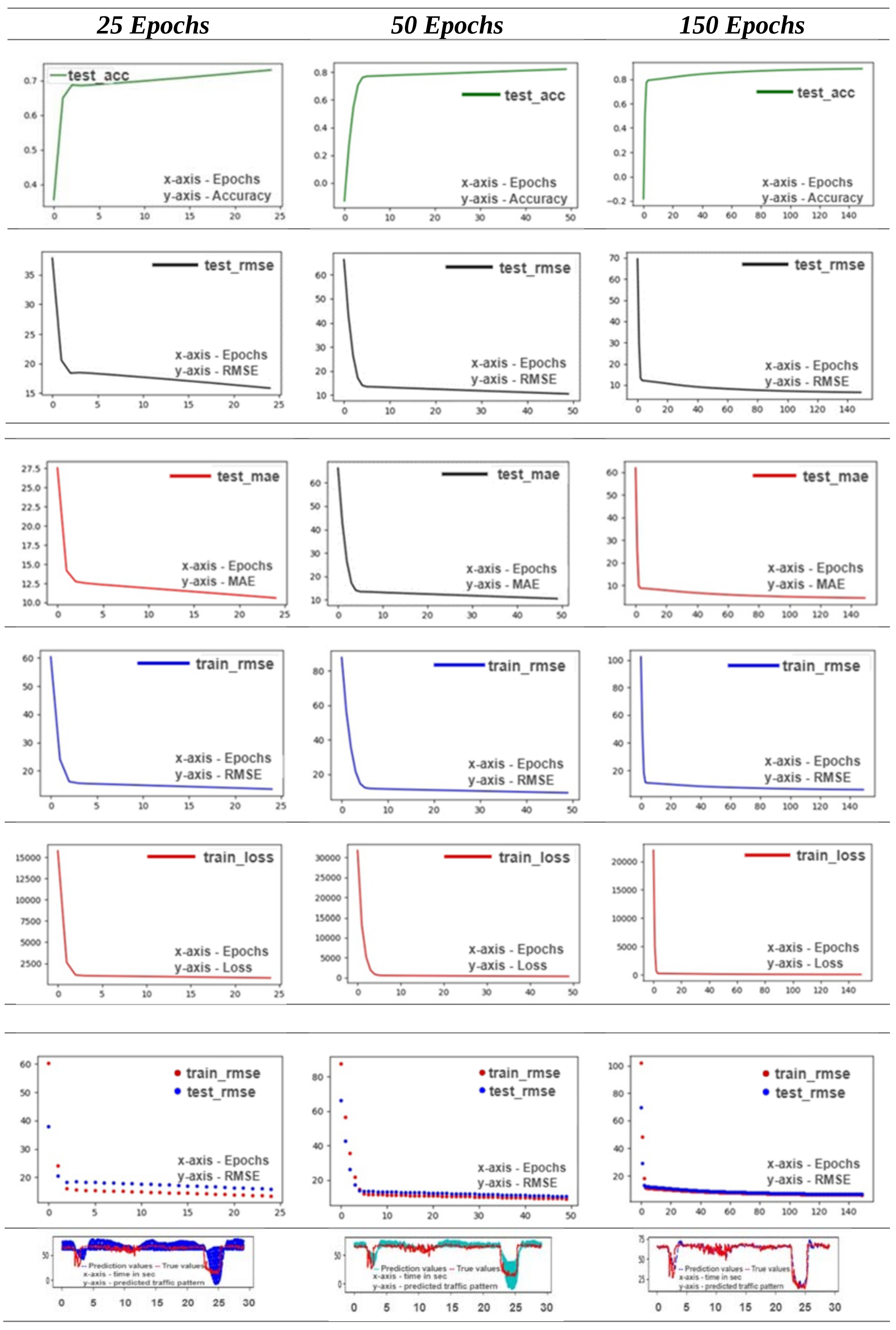

Fig. 4 *Comparing the performance of dilated recurrent graph neural network model trained for LR 10^{-5} for 25, 50 and 150 epochs.*

Temporal series	Model	Evaluation metrics					
factor		RMSE	MAE	acc	R^2	var	Loss
T	DBSCAN	5.667	3.231	0.794	0.701	0.690	*
	HA	7.442	4.014	0.873	0.712	0.712	*
	ARIMA	10.043	7.683	0.827	0.002	*	*
	GCN	7.792	5.352	0.867	0.684	0.684	*
	DRGNN	5.896	4.234	0.856	0.702	0.665	0.0012
$T+k$	DBSCAN	5.667	3.231	0.794	0.701	0.690	*
	HA	7.442	4.014	0.873	0.712	0.712	*
	ARIMA	9.345	7.689	0.827	0.003	*	*
	GCN	8.335	5.611	0.858	0.640	0.640	*
	DRGNN	5.962	4.123	0.878	0.762	0.732	0.0032
$T+2k$	DBSCAN	5.667	3.231	0.794	0.701	0.690	*
	HA	7.442	4.014	0.873	0.712	0.712	*
	ARIMA	10.053	7.695	0.827	*	0.003	*
	GCN	9.265	6.289	0.842	0.633	0.559	*
	DRGNN	6.025	4.082	0.899	0.821	0.814	0.0026

Tab. II *Comparison between state-of-the-art models with DRGNN.*

memory and computation power. The spatial, temporal and the spatio-temporal aspect of the road network could be determined using the graph based recurrent neural network, which helps in precise traffic monitoring and prediction.

5. Conclusion

The social network of things, being widely used and also holding interdisciplinary aspects of multiple revolutionary technological systems, an efficient method for analyzing the network is required underway. Thus, our proposed model aims to integrate parametric factors associated with social IoT networks namely the static traffic information and dynamic external parameters concerned with the traffic based social IoT data. The urban road network has been modelled using a graph network, where the nodes of the graphs represent the roads and the edges reflect the connections between the roadways, and the attributes of the nodes are depicted to be the traffic information on the roads with the spatio-temporal factor taken into consideration. The robustness and the accuracy of the model are analysed and evaluated. The proposed model could be able to capture the spatial and temporal features from traffic data and does not limit itself from just forecasting of traffic, but rather be applied to other aspects of spatio-temporal tasks involved in any other social IoT networks since our model is dynamic and scalable. DRGNN model with a learning rate of 10^{-5} at 150 epochs attained the maximum accuracy of 89% and outperformed the state-of-the-art models in predicting the traffic pattern. In future, varying the dilation rate could be tried on different real time datasets to make informed traffic analysis.

References

[1] MALEKSHAHI RAD M., RAHMANI A.M., SAHAFI A., NASIH QADER N. Social Internet of Things: vision, challenges, and trends. Human-centric Computing and Information Sciences, 2020, 10(1), pp. 1–40, doi: 10.1186/s13673-020-00254-6.

[2] DONG G., TANG M., WANG Z., GAO J., GUO S., CAI L., GUTIERREZ R., CAMPBELL B., BARNES L.E., BOUKHECHBA M. Graph Neural Networks in IoT: A Survey. arXiv preprint arXiv:2203.15935, 2022, 29, doi: 10.48550/arXiv.2203.15935.

[3] LIU X., LIANG Y., ZHENG Y., HOOI B., ZIMMERMANN R. Spatio-temporal graph contrastive learning. arXiv preprint arXiv: 2108.11873, 2021, doi: 10.48550/arXiv.2108.11873.

[4] ZHAO Q., ZE Y., CHAO C., YUSU W. Persistence enhanced graph neural network. In International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 2896–2906.

[5] WANG R., NGUYEN D.D., WEI G.W. Persistent spectral graph. International journal for numerical methods in biomedical engineering, 2020, 36(9), e3376, doi: 10.1002/cnm.3376.

[6] CASAS S., GULINO C., LIAO R., URTASUN R. Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data. In: IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 9491–9497.

[7] XIE Z., LV W., HUANG S., LU Z., DU B., HUANG R. Sequential graph neural network for urban road traffic speed prediction. IEEE Access. 2019, 8, pp. 63349–58, doi: 10.1016/j.eswa.2022.117921.

[8] PENG H., WANG H., DU B., BHUIYAN M.Z., MA H., LIU J., WANG L., YANG Z., DU L., WANG S., PHILIP S.Y. Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting. Information Sciences, 2020, 521, pp. 277–90, doi: 10.1016/j.ins.2020.01.043.

[9] DIEHL F., BRUNNER T., TRUONG LE M., KNOLL A. Graph neural networks for modelling traffic participant interaction. In: *IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 695–701.

[10] GAO W., FANG Y., ZHANG F., YANG Z. Representation learning of knowledge graphs using convolutional neural networks, *Neural Network World*, 2020, 30(3), pp. 145–160, doi: 10.14311/NNW.2020.30.011.

[11] KRANZ M., ROALTER L., MICHAHELLES F. Things that twitter: social networks and the internet of things. In What can the Internet of Things do for the Citizen (CIoT) workshop at the eighth international conference on pervasive computing (Pervasive 2010), 2010, pp. 1-10.

[12] JIANG L., LIU L., YAO J., SHI L. A user interest community evolution model based on subgraph matching for social networking in mobile edge computing environments. Journal of Cloud Computing. 2020, 9(1), pp. 1–5, doi: 10.1186/s13677-020-00217-3.

[13] WANG T., TAN J., DING W., ZHANG Y., YANG F., SONG J., HAN Z. Intercommunity detection scheme for social Internet of Things: Compressive sensing over graphs approach. IEEE Internet of Things Journal. 2018, 15, 5(6), pp. 4550–7, doi: 10.1109/JIOT.2018.2837048.

[14] PERERA C., ZASLAVSKY A., CHRISTEN P., GEORGAKOPOULOS D. Sensing as a service model for smart cities supported by internet of things. Transactions on emerging telecommunications technologies. 2014, 25(1), pp. 81–93, doi: 10.1002/ett.2704.

[15] HU L., WU G., XING Y., WANG F. Things2vec: Semantic modeling in the internet of things with graph representation learning. IEEE Internet of Things Journal. 2019, 27, 7(3), pp. 1939–48, doi: 10.1109/JIOT.2019.2962630.

[16] FRIJI H., KHANFOR A., GHAZZAI H., MASSOUD Y. An End-to-End Smart IoT-Driven Navigation for Social Distancing Enforcement. IEEE Access. 2022, 21, 10, pp. 76824–41, doi: 10.1109/access.2022.3192860.

[17] DEFFERRARD M., BRESSON X., VANDERGHEYNST P. Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems. 2016, 29, doi: 10.48550/arXiv.1606.09375.

[18] ZHANG S., TONG H., XU J., MACIEJEWSKI R. Graph convolutional networks: a comprehensive review. Computational Social Networks. 2019, 6(1), pp. 1–23, doi: 10.1186/s40649-019-0069-y.

[19] WANG J., KIM J., MOON S., KIM S., PARK S., PARK C.S. Spatial data dependence graph simulator for convolutional neural network accelerators. In: *IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, IEEE, 2019, pp. 309–310.

[20] LEA C., FLYNN M.D., VIDAL R., REITER A., HAGER G.D. Temporal convolutional networks for action segmentation and detection. In proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 156–165.

[21] KHANFOR A., NAMMOUCHI A., GHAZZAI H., YANG Y., HAIDER M.R., MASSOUD Y. Graph neural networks-based clustering for social internet of things. In: *IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)*, IEEE, 2020, pp. 1056–1059.