Dynamic SIoT Network Status Prediction

Dong Hu[1], Shuai Lyu[2], Shih Yu Chang[3], Limei Peng[1,4,*] and Pin-Han Ho[5]
[1]Department of Data Convergence Computing, Kyungpook National University, Daegu 41566, South Korea, e-mail: hudong@knu.ac.kr
[2]Department of Computer Science and Engineering, Kyungpook National University, Daegu 41566, South Korea, e-mail: chanceuxshuai@knu.ac.kr
[3]Department of Applied Data Science, San Jose State University, San Jose, CA 95192, USA , e-mail: shihyu.chang@sjsu.edu
[4]Department of Computer Science and Engineering, Kyungpook National University, Daegu 41566, South Korea. e-mail: auroraplm@knu.ac.kr, *Corresponding author
[5]Department of Electrical and Computer Enginnering, University of Waterloo, Waterloo, ON, N2L 3G1, Canada, e-mail: p4ho@uwaterloo.ca

Prediction of social IoT (SIoT) data traffic is helpful in characterizing the complicated relationships for such as device-to-device, user-to-user, and user-to-device. One of the most popular traffic prediction methods in noisy environments is the Kalman filter (KF), which is extremely simple and general. Nevertheless, KF requires a dynamic traffic and measurement model as a priori, which introduces extra overhead and is often difficult to obtain in reality. In comparison, deep learning models with a Recurrent Neural Network (RNN) structure have been used extensively in modeling dynamic models evolving over time. On the other hand, the Content Adaptive Recurrent Unit (CARU) is an improvement of RNN that uses fewer parameters than the LSTM and GRU and thus is more promising in predicting the SIoT data traffic. This paper proposes the CARU-based extended Kalman filter (CARU-EKF) model, which is a new deep learning cell that utilizes CARU to predict extended Kalman filter (EKF) system parameters. Note that EKF is proper to predict nonlinear SIoT traffic in noisy environments. The proposed CARU-EKF can improve the performance of time-series data forecasting for nonlinear SIoT data traffic. Numerical experiments are conducted to evaluate the SIoT traffic prediction performance of the proposed CARU-EKF approach over two real datasets, i.e., IoT device traffic and wikipedia webpage visiting traffic. The proposed method shows better performance than existing prediction methods in terms of metrics of Mean absolute error (MAE), mean absolute percentage error (MAPE), root mean square error (RMSE) and determination coefficient (R^2).

***Index Terms*—Extended Kalman filter (EKF), video frame-size predictor, multimedia network, MPEG-4 codec, matrix-based Levenberg-Marquardt algorithm (MLMA), normalized mean square error (NMSE).**

I. Introduction

The Internet of things (IoT) is an interconnected system of computing devices, machinery, and digital machines that digitize the real world. The IoT has already affected people's lives, including transportation, housing, food, clothing, health, and remote monitoring. Many home appliances can be controlled through mobile phones and voice. Many applications allow users to improve their quality of life and even enable the elderly and the disabled to live more conveniently. MGI's report shows that starting from 2025, the Internet of Things will create an output value of 3.9 trillion to 11.1 trillion US dollars in nine environments, including factories, retail, and cities, and the number of IoT devices is expected to grow to 754 100 million, which is equivalent to adding 127 IoT devices every second in the world starting in 2020 [1]. In the next generation of IoT, the objects are integrated with our social dimension, making them smart and social objects. Social Internet of Things (SIoT) is a new concept of combining such social network relations with IoT [2]. Under such development, we have found a trend of social relationships explosion which includes device-device relationships, user-user relationships, and user-device relationships. To realize the social interactions, objects should start establishing social relationships based on the object profile, shared hobbies, and social activities. As shown by Fig. 1, these social interactions can be classified into the following types: a 'co-location' relationship, i.e., objects are close in geographical location; a 'co-work' relationship, i.e., objects work together for some work; a 'parental' relationship, i.e., objects have common device attributes, like devices model; a 'social' relationship, i.e., objects are associated by social interactions; and a 'co-ownership' relationship, i.e., objects are owned by the same users.

Manuscript received April 10, 2022; revised July 10, 2022. Corresponding author: Limei Peng (email: auroraplm@knu.ac.kr)

However, the increasing number of these relationships and their heterogeneous social attributes have introduced various computing and communication challenges that prevent the IoT network from taking advantage of these relationships to improve the offered services and personalize the delivered content. It is helpful to forecast the SIoT data traffic to predict the above relationship and social behavior. For example, a network traffic-aware mobile application recommendation system based on social network interaction traffic is proposed [3]. We also can understand further about users' social behaviors by predicting web traffic generated by a user [4]. However, there are very few works dedicated in this direction.

Autoregressive Integrated Moving Average (ARIMA) is a common machine learning approach [5] used for traffic

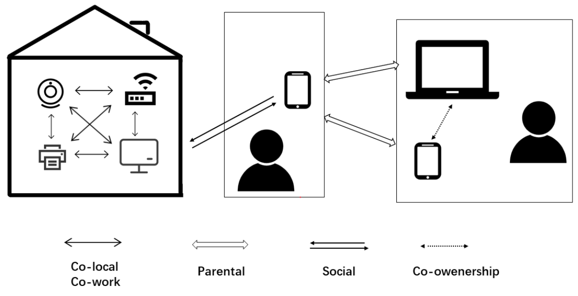

Fig. 1. Types of social relationships among objects.

prediction. In [6], the authors try to combine ARIMA prediction scheme with Generalized Auto Regressive Conditional Heteroskedasticity (GARCH) and show that it provides better prediction accuracy than ARIMA method. In order to make prediction over long-range, ARIMA is also combined with the Markov Modulated Poisson process (MMPP) [7]. All these conventional models require a large extent on the stationary data used for prediction and the prediction time range. Therefore, these models are only able to be trained by a particular time series, and they are not easy to be generalized to predict a set of time-series with a single model training. Deep learning based models are proper to solve this problem since deep learning methods are proper to deal with nonlinear characteristics of traffic data. Recurrent Neural Network (RNN) is first being applied to network traffic prediction by work [8]. RNNs are particularly proper for tasks that involve time series. In practice, Long short-term memory (LSTM) model is often used, as opposed to a vanilla (or standard) RNN, because it is more computationally effective. In fact, the LSTM was introduced to solve a problem that standard RNNs suffer from [9]. Recently, Content Adaptive Recurrent Unit (CARU) is a variant of GRU, introduced in 2020 by Ka-Hou Chan et al [10]. The CARU contains the update gate like GRU, but introduces a content-adaptive gate instead of the reset gate and it is shown that the CARU cell has fewer parameters than the LSTM and GRU.

The other issue faced by SIoT traffic prediction problem is how to perform prediction over a noisy environment. Because aforementioned machine learning or deep learning based methods have to assume that all training data are clean enough to extract traffic information, however, this assumption is not practical in practical SIoT environment. Extended Kalman Filter (EKF) is an important method to learng true states based on noisy observations and its application to traffic prediction can be found in [11]. Based on the above observation, this paper proposes using CARU-extended Kalman filter (CARU-EKF), a new deep learning cell that is proper to predict nonlinear SIoT traffic at noisy environment. In order to verify the effectiveness of the proposed CARU-EKF, numerical experiments are conducted to evaluate the SIoT traffic prediction performance of the proposed new CARU-EKF method based on two real datasets: IoT device traffic and wikipedia webpage visiting traffic. The proposed method has better performance than exisiting prediction methods by using the following performance metrics: the mean absolute error (MAE), the mean absolute percentage error (MAPE), the root mean square error (RMSE) and the coefficient of determination (R^2).

The rest of this paper is organized as follows. The essential technical components of the proposed model, extended Kalman filter and content-adaptive recurrent unit, are introduced in Section II. In Section III, we will discuss the proposed CARU-EKF prediction model. The prediction performance evaluation of the proposed CARU-EKF prediction model in terms of MAE, MAPE, RMSE and R^2 is provided by Section IV. Finally, conclusion will be drawn in Section V.

II. Model Components

In this section, we will review construction components required to build the proposed CARU-EKF model. The Kalman filter (KF) and Extended Kalman filter (EKF) will be introduced in Section II-A. The neural unit CARU will be reviewed in Section II-B.

A. Kalman Filter and Extended Kalman Filter

For dynamic and control theory, Kalman filtering, is an algorithm that uses a series of measurements observed over time, including background noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone, by estimating a joint probability distribution over the variables for each timeframe. The filter is named after Rudolf E. Kálmán, who was one of the primary developers of its theory.

Mathematically, if we represent our state, i.e., the real-time state data at time t, as vector x_t, and our measurement, i.e., the error data obtained through observation at time t, as z_t, the state dynamics of KF can be formulated as

$$x_{t+1} = \Phi x_t + w_t, \tag{1}$$

where w_t is the associated white noise process with known covariance Q. The observation dynamics of KF can be formulated as

$$z_t = H x_t + v_t. \tag{2}$$

The covariances of the two noise models are assumed stationary over time and are given by

$$Q = E\left[w_t w_t^T\right], \tag{3}$$

and

$$R = E\left[v_t v_t^T\right]. \tag{4}$$

The error covariance matrix at time t can be defined as

$$P_t = E\left[(x_t - \hat{x}_{t|t})(x_t - \hat{x}_{t|t})^T\right], \tag{5}$$

where $\hat{x}_{t|t}$ is the estimated state at time t. We assume that the prior estimate of $\hat{x}_{t|t}$ is denoted as $\hat{x}_{t|t-1}$, the relationship between $\hat{x}_{t|t}$ and $\hat{x}_{t|t-1}$ can be expressed as

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t\left(z_t - H\hat{x}_{t|t-1}\right), \tag{6}$$

where K_t is the Kalman gain.

The expression $z_t - H\hat{x}_{t|t-1}$ in Eq. (6) is known as the *innovation residual*, which can be expressed as

$$i_t = z_t - H\hat{x}_{t|t}. \tag{7}$$

By applying Eq. (7) into Eq. (6), we will get

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t\left(Hx_t + v_t - H\hat{x}_{t|t-1}\right). \tag{8}$$

Similarly, by applying Eq. (8) into Eq. (5), we obtain

$$\begin{aligned} P_t = E \quad & \left[\left[(I-K_tH)\left(x_t - \hat{x}_{t|t-1}\right) - K_tv_t\right]\right. \\ & \left.\left[(I-K_tH)\left(x_t - \hat{x}_{t|t-1}\right) - K_tv_t\right]^T\right] \end{aligned} \tag{9}$$

Since $x_t - \hat{x}_{t|t-1}$ is the error of the prior estimate, we can express Eq. (9) as

$$\begin{aligned} P_t = & (I-K_tH)\, E\left[\left(x_t - \hat{x}_{t|t-1}\right)\left(x_t - \hat{x}_{t|t-1}\right)^T\right] \\ & \cdot (I-K_tH) + K_tE\left[v_tv_t^T\right]K_t^T \end{aligned} \tag{10}$$

By applying Eqs. (4) and (5) into Eq. (9), we obtain the following:

$$P_t = (I-K_tH)\,P_t'\,(I-K_tH)^T + K_tRK_t^T, \tag{11}$$

where P_t' is the prior estimate of P_t.

In summary, we have the following update about Kalman filter

$$\begin{aligned} K_t &= P_t'H^T\left(HP_t'H^T + R\right)^{-1} && \text{(Kalman Gain)} \\ \hat{x}_{t|t} &= \hat{x}_{t|t-1} + K_t(z_t - H\hat{x}_{t|t-1}) && \text{(Post. State Update)} \\ P_t &= (I-K_tH)P_t' && \text{(Cov. Update)} \\ \hat{x}_{t+1|t} &= \phi\hat{x}_{t|t} && \text{(Pri. State Update)} \\ P_{t+1} &= \phi P_t\phi^T + Q && \text{(Cov. Update)} \end{aligned} \tag{12}$$

If the state transit relations Φ and H in Eqs. (1) and (2) do not follow linear mapping relations, we have to consider extended KF. For nonlinear mapping relations, EKF can be formulated as

$$x_{t+1} = f(x_t, u_t) + w_t, \tag{13}$$

where $f(\ )$ is a nonlinear function for the previous state and control variable $u(t)$. The observation dynamics of KF can be formulated as

$$z_t = h(x_t) + v_t, \tag{14}$$

where $h(\ )$ is a nonlinear function for the observation. To deal with EKF, we can perform the following linearization to get parameters Φ and H in Eqs. (1) and (2) as:

$$\Phi_t \approx \left.\frac{\partial f}{\partial x}\right|_{x_t}, \tag{15}$$

and

$$H_t \approx \left.\frac{\partial h}{\partial x}\right|_{x_t}. \tag{16}$$

B. Content-Adaptive Recurrent Unit

Content Adaptive Recurrent Unit (CARU) is a variant of GRU, introduced in 2020 by Ka-Hou Chan et al. [10]. The CARU includes the update gate like GRU, but introduces a content-adaptive gate instead of the reset gate. Same as GRU, the purpose of CARU is to resolve the long-term dependence problem of RNN models. It is observed that CARU can have a slight performance improvement on NLP tasks by using fewer parameters than the GRU.

We have to introduce the following parameters in order to explain CARU.

- $v^{(t)}$: the input signal at time t.
- $x^{(t)}$: It first projects the current signal $v^{(t)}$ into $x^{(t)}$ as the input feature. This result will be used in the next hidden state and passed to the proposed content-adaptive gate.
- $n^{(t)}$: This parameter is obtained by combining the parameters related to $h^{(t)}$ and $x^{(t)}$ to produce a new hidden state $n^{(t)}$.
- $z^{(t)}$: This is the update gate and is used to represent the transition of the hidden state.
- $h^{(t+1)}$: The next hidden state is combined with $h^{(t)}$ and $n^{(t)}$.
- $l^{(t)}$: This is used as the content-adaptive gate, which will influence the amount of gradual transition, rather than diluting the current hidden state.

Following equations are used to represent CARU operations. $[W;B]$ represents the training parameters, which are linear layers consisting of weights and biases, respectively. We set $t = 0$ as our initial state. CARU directly returns $h^{(1)} \leftarrow W_{vn}v^{(0)} + B_{vn}$ at the initial state. Next, for $t > 0$, a complete recursion loop for CARU is given by:

$$x^{(t)} = W_{vn}v^{(t)} + B_{vn} \tag{17}$$

$$n^{(t)} = \phi\left(\left(W_{hn}h^{(t)} + B_{hn}\right) + x^{(t)}\right) \tag{18}$$

$$z^{(t)} = \sigma\left(W_{hz}h^{(t)} + B_{hz} + W_{vz}v^{(t)} + B_{vz}\right) \tag{19}$$

$$l^{(t)} = \sigma\left(x^{(t)}\right) \odot z^{(t)} \tag{20}$$

$$h^{(t+1)} = \left(1 - l^{(t)}\right) \odot h^{(t)} + l^{(t)} \odot n^{(t)} \tag{21}$$

Note that it has $t \leftarrow t+1$ at the end of each recurrent loop. The operator $\odot$ denotes the Hadamard product. σ and ϕ are the activation functions of sigmoid and hyperbolic tangent, respectively.

There are three pipelines of data flow to be processed by CARU. The first pipeline is about *content-state*. It produces a new hidden state $n^{(t)}$ achieved by a linear layer, this part is equivalent to simple RNN networks. The second pipeline is about *signal-state*. It produces the weight $\sigma(x^{(t)})$ of the current signal. From a function perspective, it has the similar function to a GRU reset gate but is only based on the current signal instead of the entire signal. More specifically, it can be considered as the tagging task that connects the relation between the weight and parts-of-speech. The third pipeline is *content-weight*. It produces the weight $z^{(t)}$ of the current content, the form is the same as a GRU update gate and the purpose for this data flow pipeline is to overcome the long-term dependence.

III. CARU-EKF Prediction Model

In this section, we will utilize aforementioned two construction components EKF and CARU to present the proposed CARU-EKF model.

A. State Prediction and State Update

In this section, we present our model by utiliznig the Content Adaptive Recurrent Unit combined with extended Kalman filter (CARU-EKF) for SIoT traffic prediction. The main idea is to utilize CARU units to build a neural network model that can model nonlinear functions f and h given by Eq. (13) and Eq. (14). Under this way, we can apply Kalman filters without the requirement to specify a linear transition functions Φ and H or fixed process and measurement covariance matrices $\mathbf{Q}$ and $\mathbf{R}$. Instead, we will model a nonlinear transition function f and h along with $\mathbf{Q}$, and $\mathbf{R}$ using different CARU neural networks. Such proposed model enables us to learn nonlinear dynamics from traffic data.

Since we always assume that observations and measurements are noisy estimates of the underlying state, the EKF for the SIoT traffic prediction can be formulated as

$$x_{t+1} = f(x_t) + w_t, \tag{22}$$

where $f(.)$ is a nonlinear function for the previous traffic state. The observation dynamics of EKF can be formulated as

$$z_t = h(x_t) + v_t, \tag{23}$$

where $h(.)$ is a nonlinear function for the observation. Eqs. (22) and (23) are used to specify the underlying model of the CARU-EKF. The covariances of the two noise models become dynamic now, they can be expressed as

$$Q_t = E\left[w_t w_t^T\right], \tag{24}$$

and

$$R_t = E\left[v_t v_t^T\right]. \tag{25}$$

According to Section II-A, the prediction step is

$$\begin{aligned} \hat{x}_{t|t} &= f(\hat{x}_{t|t-1}), \\ P_{t+1} &= \phi_t P_t \phi_t^T + Q_t \end{aligned} \tag{26}$$

where f is modeled by one CARU module, ϕ_t is the derivative of f with respect to $\hat{x}_{t|t-1}$. The covariance matrix Q_t is also generated by another CARU model. The EKF recursion update step is

$$K_t = P_t' H^T \left(H P_t' H^T + R_t\right)^{-1}, \tag{27}$$

$$\hat{x}_{t|t} = \hat{x}_{t|t} + K_t(z_t - H_t \hat{x}_{t|t-1}), \tag{28}$$

$$P_t = (I - K_t H) P_t'. \tag{29}$$

where $\mathbf{R}_t$ is the output of a third CARU module and where z_t is our observed measurement at time t .

B. Model Structure

We will use CARU to model f, $\mathbf{Q}_t$, and $\mathbf{R}_t$ by considering input z_{t-1}, which is the network status vector obtained by variational autoencoder from the network status. After CARU trains required parameters for EKF, it will be used to generate the output $\hat{x}_{t|t}$. The overview of the CARU-EKF unit is depicted in part (a) in Fig. 2.

At each time step t, CARU$_f$ takes in the previous prediction $\hat{x}_{t-1|t-1}$ as input and produces the intermediate state $f(\hat{x}_{t-1|t-1})$. Note that this term is indepedent on the current measurement. The CARU$_Q$ will takes $f(\hat{x}_{t-1|t-1})$ as input and produces an estimate of the process covariance, Q_t , as output. Similarly, the observation z_{t-1} from the network status serves as input to CARU$_R$, which only produces an estimate of the measurement covariance, R_t, as output. Finally, $f(\hat{x}_{t-1|t-1})$ and z_{t-1}, along with our covariance estimates Q_t and R_t , are fed to a extended Kalman filter for update at time t discussed by Section III-A. The CARU-EKF unrolled over time, which can be trained end to end with backpropagation through time as shown by part (b) in Fig. 2.

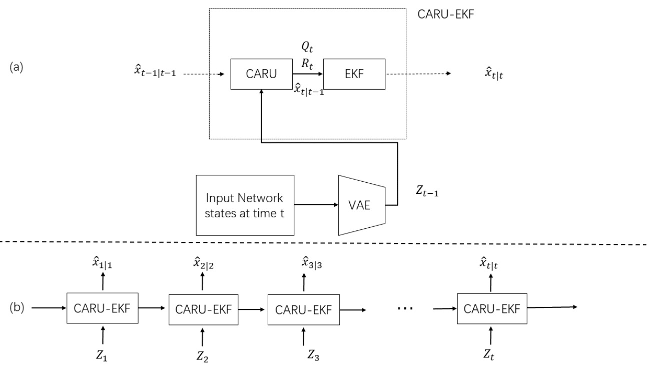

Fig. 2. Structure of the proposed CARU-EKF mode. (a) The CARU-EKF cell details. (b) The CARU-EKF unrolled over time, which can be trained end to end with backpropagation over time.

1) Loss function of models

The loss function for neural network CARU$_f$ used to train nonlinear f can be expressed as

$$L(\theta_f) = \frac{1}{T} \sum_{t=1}^{T} \left\| x_{t|t} - \hat{x}_{t|t}(\theta_f) \right\|^2 + \lambda_f \left\| \text{CARU}_f \right\|^2, \tag{30}$$

where $\hat{x}_{t|t}(\theta_f)$ is the output of the neural network CARU$_f$, λ_f is the regularization parameter and $\|\text{CARU}_f\|^2$ are squared summations of neural network CARU$_f$ model weights.

The loss function for neural network CARU$_Q$ used to train Q_t can be expressed as

$$L(\theta_Q) = \frac{1}{T} \sum_{t=1}^{T} \left\| Q_t - \hat{Q}_t(\theta_Q) \right\|^2 + \lambda_Q \left\| \text{CARU}_Q \right\|^2, \tag{31}$$

where $\hat{Q}_t(\theta_Q)$ is the output of the neural network CARU$_Q$, λ_Q is the regularization parameter and and $\|\text{CARU}_Q\|^2$ are squared summations of neural network CARU$_Q$ model weights.

The loss function for neural network CARU$_R$ used to train R_t can be expressed as

$$L(\theta_R) = \frac{1}{T}\sum_{t=1}^{T}\left\|R_t - \hat{R}_t(\theta_R)\right\|^2 + \lambda_R \left\|\mathrm{CARU}_R\right\|^2, \quad (32)$$

where $\hat{R}_t(\theta_R)$ is the output of the neural network CARU$_R$, λ_R is the regularization parameter and and $\|\mathrm{CARU}_R\|^2$ are squared summations of neural network CARU$_R$ model weights.

2) Optimization

Our objective is to optimize all wegiths θ_f, θ_Q and θ_R by minimizing the loss given by Eqs (30), (31) and (32) with respect to all free weights in our model, which are a concatenation of all weight matrices and biases from all three CARU models. Our CARU-EKF is trained end to end, with gradients obtained using the backpropagation through time algorithm, which we implement using the TensorFlow Keras framework with gradient updates according to the Adam optimizer.

IV. Numerical Experiments

In this section, we will apply the proposed CARU-EKF model to predict two SIoT related datasets. The first one is an IoT traffic dataset collected from a mobile operator [12] and the investigation of this dataset will be given in Section IV-A. In Section IV-B, we will apply CARU-EKF to predict wikipedia webpages traffic based on web traffic time series dataset provided by [13]. Following metrics are adopted in our prediction performance evaluation: the mean absolute error (MAE), the mean absolute percentage error (MAPE), the root mean square error (RMSE) and the coefficient of determination (R^2). These metric can be defined as

$$\begin{aligned} \mathrm{MAE} &= E(|Y-\hat{Y}|); \\ \mathrm{MAPE} &= E\left((Y-\hat{Y})/Y\right)\times 100\%; \\ \mathrm{RMSE} &= \sqrt{E((Y-\hat{Y})^2)}; \\ R^2 &= 1-\frac{\sum_{i=0}^{N}\left(Y_i-\hat{Y}_i\right)^2}{\sum_{i=0}^{N}\left(Y_i-E(Y)\right)^2}, \end{aligned} \quad (33)$$

where $\hat{Y}, \hat{Y}_i$ are pedicted values and Y, Y_i are true (actual) values.

A. Performance Evaluation for IoT Traffic Dataset

The dataset used in this section consists of activity records in units of bytes for both transmission and reception for 6214 mobile devices during a consecutive period of 630 hours (about 26.25 days). The traffic size per mobile device has been combined in a time window of one hour long. Besides the traffic status, this dataset also includes mobile identification information (SIM) and the timestamp of traffic records. We partition the dataset into training and test parts. The training parts are collected for the first 500 hours and the test dataset is obtained from the remaining 130 hours.

There are two issues with the current dataset. The first is about the unbalanced nature of the distribution of traffic. In order to handle this issue, we apply the following methods. The first method is to under-sample the dataset with the abundant categories. This method is proper to be applied to the dataset when the quantity of data is sufficient. By reserving all data in the rare categories and randomly selecting an equal number of data in the abundant categories, a balanced dataset can be obtained for later training. When some datasets are insufficient of quantity data, the second method is to try to increase the size of data collected from rare categories. New data collected from rare categories can be generated by repetition, simulation, augmentation, bootstrapping or SMOTE (Synthetic Minority Over-Sampling Technique). We have to pay attention that there is no absolute advantage of one resampling method over another. Application of these methods depends on the application scenarios it applies to and the dataset itself. The third method is to combine different re-sampled datasets. The simplest way to successfully generalize a model is by utilizing more data. The problem is that out-of-the-box models like logistic regression or xgboost tree tend to generalize by discarding the rare category. One practical way is to build m models that use all the samples of the rare categories and m-differing samples of the categories with plentiful data. Given that one wishes to ensemble 100 models, one would keep e.g. the 10000 cases of the rare class and randomly sample 100000 cases of the abundant class. Then one just split the 100000 cases in 100 chunks and trained 100 different models. The last method we try to deal with an imbalanced dataset is to resample with different ratios. The previous methods can be fine-tuned by changing with the ratio between the rare and the abundant categories. The best ratio will depend on the data and the models that are used. Besides training all models with the same ratio in the ensemble, it is also possible to try the ensemble for different ratios. For example, if 5 models are trained, it might make sense to have a model that has a ratio of 1:1 (abundant:rare) and another one with 2:1, or even 3:1. The first challenge is more difficult and brings us more extra data preprocessing tasks before applying our prediction algorithm.

The other challenge of this traffic dataset is the wide value ranges for the traffic status. The value range can be broad as several orders of magnitude. We deal with this problem by applying a $\log$ transform to the traffic volumes. The $\log$ transform is based on the following formulation: $\log(y+1)$ where y is the traffic data and $\log$ is the natural logarithm.

Considering the high number of idle connections in the 1-h time slot aggregation period, it is also interesting to show the distribution of active connections (connections with a nonzero traffic volume in the 1-h period) considering the day of week (Monday to Sunday) and the hour of day (0 to 23 h). Fig. 3 provides these distributions of active connections. We observe the expected hour/day cyclic activity corresponding to the distribution of working hours/days.

The sampled traffic data for each weekday from Monday to Sunday is presented by Figure 3. If we consider the high number of idle connections in the 1-h time slot aggregation period, we also show the distribution of active connections per hour in Figure 4.

We apply our proposed new CARU-EKF approach to predict

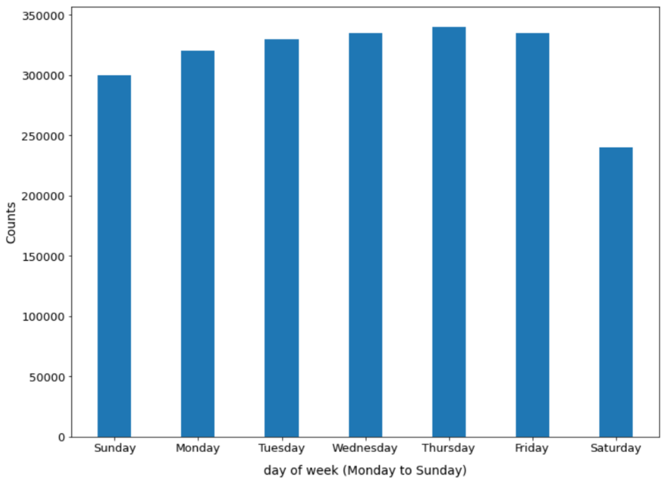

Fig. 3. Distribution of network traffic per weekday.

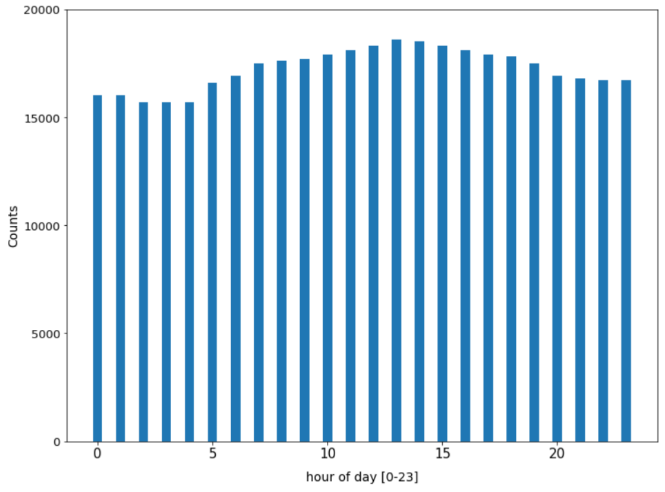

Fig. 4. Distribution of network traffic per hour.

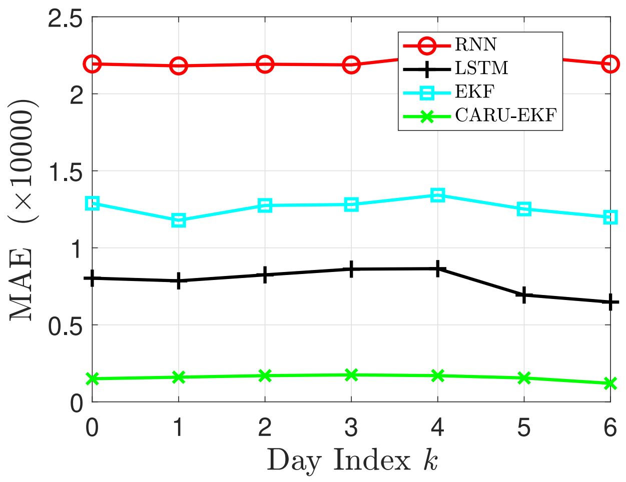

Fig. 5. The MAE performances resulting from the RNN, EKF, CARU and CARU-EKF schemes with respect to each weekday for the realworld IoT traffic data from a mobile operator.

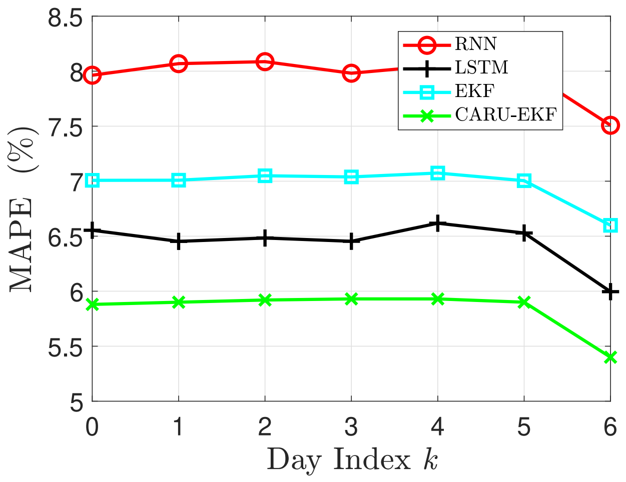

Fig. 6. The MAPE performances resulting from the RNN, EKF, CARU and CARU-EKF schemes with respect to each weekday for the realworld IoT traffic data from a mobile operator.

the traffic states $\hat{x}_{t|t}$ per weekday, i.e., t is sampled per day. Figure 5 depicts the MAEs as defined by Eq. (33), which result from our proposed new CARU-EKF method and three existing deep-learning prediction methods, namely the recurrent neural network model in [8] (denoted by "RNN" in the figures), the long-term memory network model in [9] (denoted by "LSTM" in the figures), and the extended Kalman filter method in [11] (denoted by "EKF" in the figures). In addition, Figure 6 delineates the MAPEs as defined by Eq. (33), which result from our proposed new CARU-EKF approach and the above-state three deep-learning prediction methods. Figure 7 plots the RMSEs as defined by Eq. (33), which result from our proposed new CARU-EKF prediction scheme and the above-state three deep-learning prediction methods. Finally, Figure 8 plots the R^2 as defined by Eq. (33).

In order to validate that the proposed method is also applicable to higher resolution time scale, we will compare the proposed CARU-EKF with other deep-learning prediction methods by predicting the traffic states $\hat{x}_{t|t}$ per hour, i.e., t is sampled per hour. Figure 9 depicts the MAEs as defined by Eq. (33), which result from our proposed new CARU-EKF method and three existing deep-learning prediction methods, namely the recurrent neural network model in [8] (denoted by "RNN" in the figures), the long-term memory network model in [9] (denoted by "LSTM" in the figures), and the extended Kalman filter method in [11] (denoted by "EKF" in the figures). In addition, Figure 10 plots the MAPEs as defined by Eq. (33), which result from our proposed new CARU-EKF approach and the above-state three deep-learning prediction methods. Figure 11 generates the RMSEs as defined by Eq. (33), which result from our proposed new CARU-EKF prediction scheme and the above-state three deep-learning prediction methods. Figure 8 delineates the R^2 as defined by Eq. (33) to verify the fitness of the proposed model and deep-learning prediction approaches when the samping period becomes an hour. According to Figures 9-12, the proposed new CARU-EKF method has better performance than the three existing deep-learning prediction schemes.

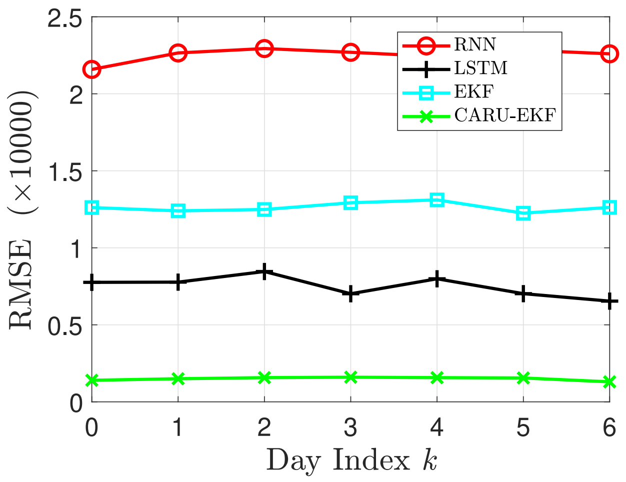

Fig. 7. The RMSE performances resulting from the RNN, EKF, CARU and CARU-EKF schemes with respect to each weekday for the realworld IoT traffic data from a mobile operator.

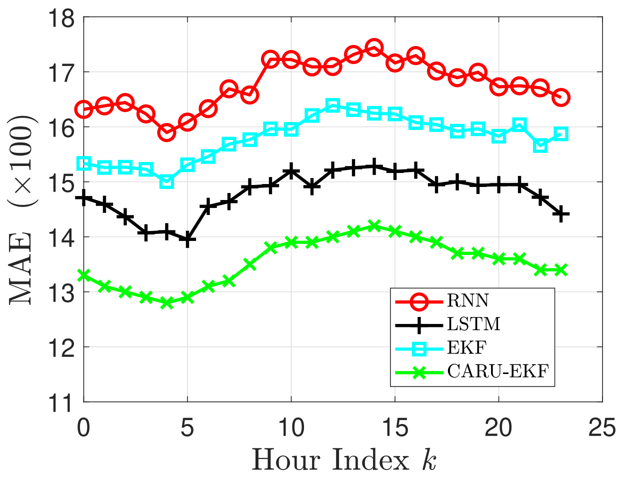

Fig. 9. The MAE performances resulting from the RNN, EKF, LSTM and CARU-EKF schemes with respect to each hour per day for the realworld IoT traffic data from a mobile operator.

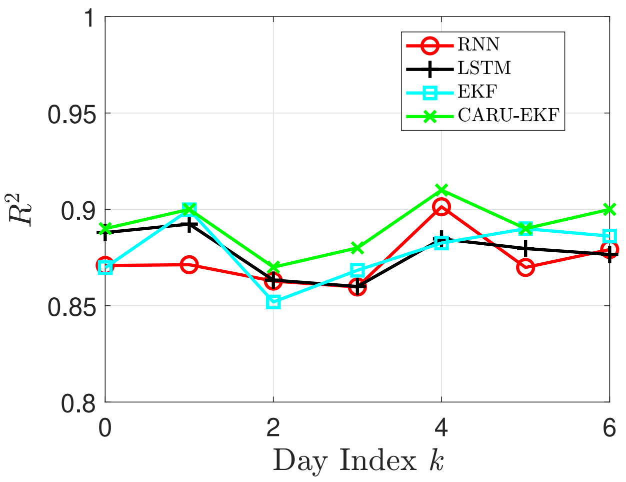

Fig. 8. The R^2 performances resulting from the RNN, EKF, CARU and CARU-EKF schemes with respect to each weekday for the realworld IoT traffic data from a mobile operator.

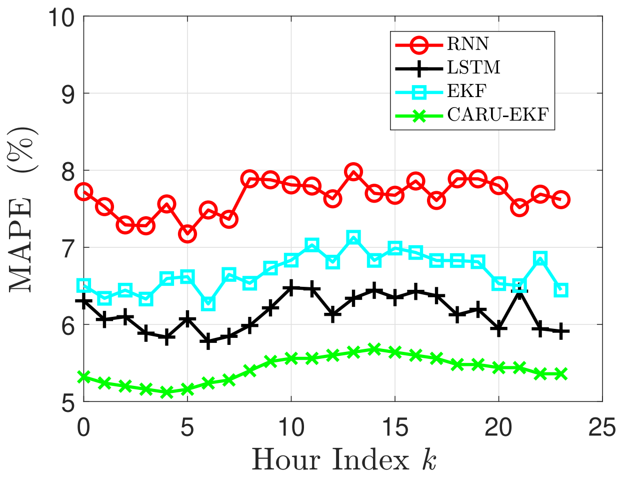

Fig. 10. The MAPE performances resulting from the RNN, EKF, LSTM and CARU-EKF schemes with respect to each hour per day for the realworld IoT traffic data from a mobile operator.

B. *Performance Evaluation for Wikipedia Traffic Dataset*

The second real dataset under investigation in this work is the wikipedia traffic [13]. The training dataset consists of approximately 145k time series. Each of these time series represent a number of daily views of a different Wikipedia article, starting from July, 1st, 2015 up until December 31st, 2016. For each time series, you are provided the name of the article as well as the type of traffic that this time series represents (all, mobile, desktop, spider). We compare the proposed CARU-EKF with other three prediction methods: RNN, EKF and LSTM used in Section IV-A by metrics MAE, MAPe, RMSE and R^2 defined by Eq. (33). In table I, we show the prediction performance based on the traffic visiting the AŃD wikipedia webpage [14]. Similarly, in table II, we also show the prediction performance based on the traffic visiting the Erni Mangold Wikipedia webpaged [15]. Both tables show that the proposed novel CARU-EKF method has better performance than the other three existing deep-learning prediction approaches in terms of MAE, MAPe, RMSE and R^2.

Figures 5-8, Figures 11-12, and Tables I-II show that by training the required parameters for EKF coupled with the noise reduction of the data by the EKF, greatly improves the prediction effect.our proposed new CARU-EKF method obviously outperforms the three existing deep-learning prediction methods

V. Conclusion

In order to predict traffic of SIoT, we propose a new prediction method, named CARU-EKF, based on Content Adaptive Recurrent Unit (CARU). Compared to LSTM and GRU, CARU requires less model parameters to train and the proposed CARU-EKF can help determine EKF parameters to perform noisy environment traffic prediction. We verify

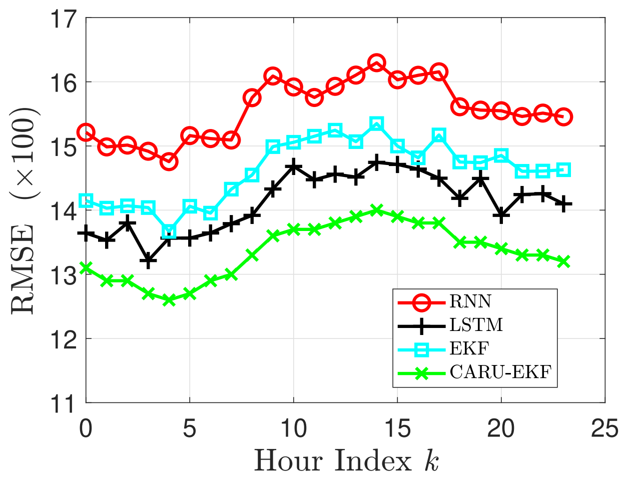

Fig. 11. The RMSE performances resulting from the RNN, EKF, LSTM and CARU-EKF schemes with respect to each hour per day for the realworld IoT traffic data from a mobile operator.

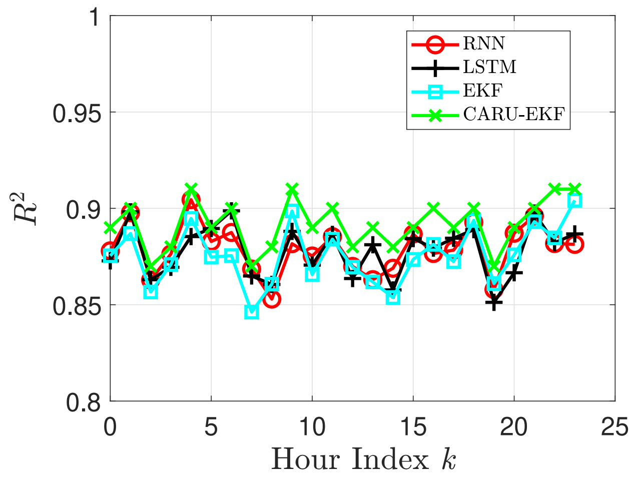

Fig. 12. The R^2 performances resulting from the RNN, EKF, LSTM and CARU-EKF schemes with respect to each hour per day for the realworld IoT traffic data from a mobile operator.

TABLE I
AŃD́ WIKIPEDIA

Model \ Performance	MAE	MAPE %	RMSE	R^2
RNN	4.2	7.8	4.4	0.88
EKF	2.3	7.0	2.4	0.87
LSTM	1.3	6.4	1.5	0.88
CARU-EKF	**0.34**	**5.8**	**0.4**	**0.89**

TABLE II
ERNI MANGOLD WIKIPEDIA

Model \ Performance	MAE	MAPE %	RMSE	R^2
RNN	5.1	7.6	5.2	0.87
EKF	3.3	7.2	3.4	0.88
LSTM	2.3	5.9	2.5	0.88
CARU-EKF	**1.34**	**5.2**	**1.36**	**0.89**

the proposed prediction method by applying CARU-EKF to two real dataset about SIoT: the first one is the IoT traffic dataset collected from mobile operators, and the second one is the traffic dataset for visiting wikipedia webpages. We use four performance metrics: the mean absolute error (MAE), the mean absolute percentage error (MAPE), the root mean square error (RMSE) and the coefficient of determination (R^2), to evaluate and compare the proposed method with RNN, LSTM and EKF prediction methods. Numerical results demonstrate the proposed CARU-EKF has better prediction performance compared to conventional methods.

VI. ACKNOWLEDGEMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (Grant No.: 2020R1I1A3072688).

REFERENCES

[1] P. Gokhale, O. Bhat, and S. Bhat, "Introduction to IOT," *International Advanced Research Journal in Science, Engineering and Technology*, vol. 5, no. 1, pp. 41–44, 2018.
[2] F. Al-Turjman, "5G-enabled devices and smart-spaces in social-IoT: an overview," *Future Generation Computer Systems*, vol. 92, pp. 732–744, 2019.
[3] X. Su, Y. Zheng, J. Lin, and X. Liu, "A network traffic-aware mobile application recommendation system based on network traffic cost consideration," *International Journal of Computational Science and Engineering*, vol. 19, no. 2, pp. 259–273, 2019.
[4] K. Xu, F. Wang, and L. Gu, "Behavior analysis of internet traffic via bipartite graphs and one-mode projections," *IEEE/ACM Transactions on Networking*, vol. 22, no. 3, pp. 931–942, 2013.
[5] H. Z. Moayedi and M. Masnadi-Shirazi, "ARIMA model for network traffic prediction and anomaly detection," in *2008 international symposium on information technology*, vol. 4. IEEE, 2008, pp. 1–6.
[6] B. Zhou, D. He, Z. Sun, and W. H. Ng, "Network traffic modeling and prediction with ARIMA/GARCH," in *Proc. of HET-NETs Conference*, 2005, pp. 1–10.
[7] A. Sang and S.-q. Li, "A predictability analysis of network traffic," *Computer networks*, vol. 39, no. 4, pp. 329–345, 2002.
[8] R. Vinayakumar, K. Soman, and P. Poornachandran, "Applying deep learning approaches for network traffic prediction," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2017, pp. 2353–2358.
[9] H. D. Trinh, L. Giupponi, and P. Dini, "Mobile traffic prediction from raw data using LSTM networks," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2018, pp. 1827–1832.
[10] K.-H. Chan, W. Ke, and S.-K. Im, "CARU: A content-adaptive recurrent unit for the transition of hidden state in NLP," in *International Conference on Neural Information Processing*. Springer, 2020, pp. 693–703.
[11] C. P. Van Hinsbergen, T. Schreiter, F. S. Zuurbier, J. Van Lint, and H. J. Van Zuylen, "Localized extended Kalman filter for scalable real-time traffic state estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 385–394, 2011.
[12] M. L. Martin, A. Sanchez-Esguevillas, and B. Carro, "Review of methods to predict connectivity of IoT wireless devices." *Ad Hoc Sens. Wirel. Networks*, vol. 38, no. 1-4, pp. 125–141, 2017.
[13] "Web traffic time series forecast," https://www.kaggle.com/c/web-traffic-time-series-forecasting, accessed: 2018-09-30.
[14] "The ańd́ wikipedia webpage," https://zh.wikipedia.org/wiki/AND.
[15] "The erni mangold wikipedia webpag," https://en.wikipedia.org/wiki/Erni_Mangold.