Communication Traffic Prediction with Continual Knowledge Distillation

Hang Li*, Ju Wang*, Chengming Hu*, Xi Chen*, Xue Liu*, Seowoo Jang†, and Gregory Dudek*
*Samsung Electronics, Canada; †Samsung Electronics, Korea (South)
{hang2.li,benny.hu}@partner.samsung.com,{j.wang1,alex.chen1, steve.liu, seowoo.jang, greg.dudek}@samsung.com

ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9838521

***Abstract*—Accurate traffic volume estimation and prediction are essential for advanced communication network functions, such as automatic operations and predictive resource allocation. Although machine learning (ML)-based approaches achieve great success in accomplishing this goal, existing approaches suffer from two drawbacks that limit their real-world applications. First, the ML-based prediction models developed in the past might be obsolete now, since the communication traffic patterns and volumes keep changing in the real world, leading to prediction errors. Second, most Base Stations (BSs) can only save a small amount of data due to the limited storage capacity and high storage costs, which prevents from training an accurate prediction model. In this paper, we propose a novel framework that adapts the prediction model to the constantly changing traffic with only a few current traffic data. Specifically, the framework first learns the knowledge of historical traffic data as much as possible by using a proposed two-branch neural network design, which includes a prediction and a reconstruction module. Then, the framework transfers the knowledge from an old (past) prediction model to a new (current) model for the model update by using a proposed continual knowledge distillation technique. Evaluations on a real-world dataset show that the proposed framework reduces the Mean Absolute Error (MAE) of traffic prediction by up to 9.62% compared to the state-of-the-art prediction methods.**

***Index Terms*—traffic prediction, knowledge distillation**

I. Introduction

The Self-Organizing Network (SON) is a critical technology for 5G and future communication networks to enable autonomous Operations, Administration, and Management (OAM). Prediction is critical in supporting SON functionalities, such as predictive resource allocation, dynamic spectrum management, and automated network slicing [1], [2]. There are twofold reasons. On the one hand, communication networks are subject to a variety of delays, including observation delays, computing delays, and actuation delays, which easily make a SON operation obsolete when deployed. On the other hand, future information has to be considered for stable SON operations to avoid oscillations and myopic decision making.

Traffic volume (or traffic for short in the rest of this paper) is a major reference variable for SON operations [3] and is one of the most fundamental metrics of communication networks. There exist numerous efforts on traffic prediction in the literature, tracing back to the GSM systems [4] and even earlier systems [5]. Traditional methods use statistics or probabilistic distributions for traffic prediction. Recently, Machine Learning (ML) based methods [6], especially Neural Networks (NNs) [7], [8], illustrate their strong power in modeling the non-linear relationship or characteristic in data, and thus they can further improve the prediction accuracy.

Although existing ML-based approaches achieve great success in traffic prediction, we argue that two drawbacks prevent their real-world applications. On one hand, the traffic patterns and volumes are always changing in the real world. As a result, the prediction models trained on previous data are unable to generalize effectively to a new situation and may suffer large prediction errors. On the other hand, traditional ML or NN models suffer from the well-known drawback in that they rely on a large amount of data for training an accurate model. However, the capacity of a BS is limited and the historical data will be discarded when new data arrives. Thus, a BS can only store a small amount of traffic data, which is inadequate to train an accurate prediction model. One of the seemingly possible solutions is collecting data from other BSs and training prediction models on these newly collected large data sets. Unfortunately, it could be quite expensive and difficult due to the hidden Operating Expenses (OPEX) and overheads including i) the bandwidth cost for migrating data into long-term storage [9], ii) the degradation of computing resource utilization due to frequent disk writes [10].

To tackle this issue, we introduce a novel framework - **C**ontinual **K**nowledge **D**istillation (**CKD**)- that adapts prediction models to the constantly changing communication traffic scenarios without the need of a large amount of data. Our design is based on two observations. First, although each BS only has a limited amount of data available for a given time period, there is a large amount of historical data that can help improve the accuracy. Second, saving models is more efficient than saving data, since it only occupies negligible storage space. Thus, instead of maintaining all historical traffic data, we only save one prediction model in a BS, which is cost-effective. Next, by updating the prediction model with a few new come data, our framework can achieve a high prediction accuracy over time.

At a high level, our CKD framework works in the following two steps. First, the framework learns the knowledge of historical traffic data as much as possible by using a proposed two-branch neural network structure. One branch employs the Encoder-Reconstruction structure to preserve the compact knowledge. Because if one model can accurately reconstruct data, it contains nearly all information from the data. The other

branch uses the Encoder-Prediction structure to achieve the goal of accurate prediction. Second, the framework updates the prediction model by using a designed continual knowledge distillation technique. Because the old model contains the knowledge derived from historical traffic data, we transfer the knowledge from the old model to a new model learned on new data. Then, the new model can incorporate the knowledge of both the old model and the new data. Technically, the prediction model is updated by minimizing the difference between (i) its predicted values and ground truth, and (ii) the predicted values of the old model and the new model.

This paper makes the following **contributions**:

- To the best of our knowledge, this is the first work to tackle the communication traffic variation issue in prediction. By adopting the advanced knowledge distillation technique, the proposed CKD framework can adapt the prediction model to the constantly changing traffic with a small amount of data.
- We conduct experiments on cellular traffic data collected throughout a year from 50 BSs in an anonymous location. Compared to the state-of-the-art, our framework reduces the prediction Mean Absolute Errors (MAEs) by up to 9.62%.

II. Design of Continual Knowledge Distillation

A. Definitions and Problem Statement

Suppose that a BS monitors traffic conditions and records a traffic data sample every time slot (e.g., 1 hour). Due to the capacity limitation on each BS, a BS can only store its traffic volume for a period of time, i.e. the past n time slots. The historical traffic data will be overwritten or deleted when new data comes. Hence, the amount of traffic data stored in one BS is too small to get an accurate prediction model.

In this paper, we use task t to denote different periods of time. Each task t contains n traffic data samples $\{s_i^t\}_{i=1}^n$. To facilitate the model to make time sequence prediction, we transform the traffic data by using a sliding window with a length of $c+1$. We define the first c samples as an input vector $x_i^t = [s_i^t, s_{i+1}^t, ..., s_{i+c}^t]$ and the last sample as the ground truth $y_i^t = s_{i+c+1}^t$, which is needed to be predicted. In this way, we produce a transformed dataset $D^t = \{(x_i^t, y_i^t)\}_{i=1}^m$ for task t, where $m = n - c$.

Problem Statement: For each BS, we aim to learn a prediction model based on the given D^t, and the model can predict the future traffic $\hat{y}_i$ according to the given data x_i.

B. Overview

To fulfill the mission of precise prediction, CKD employs the concept of continual learning based on two intuitions. (i) Although each BS has a limited amount of traffic data, there is a large amount of historical data that can be utilized to improve accuracy. (ii) Model storage is more efficient than data storage since it occupies small negligible storage. As such, we propose to save and update the prediction model throughout time.

CKD aims to continuously update the model by utilizing the knowledge from historical data. Fig. 1 presents the overview of CKD. Before the new task t arrives, we have saved an old model trained on the previous task $t-1$. This old model

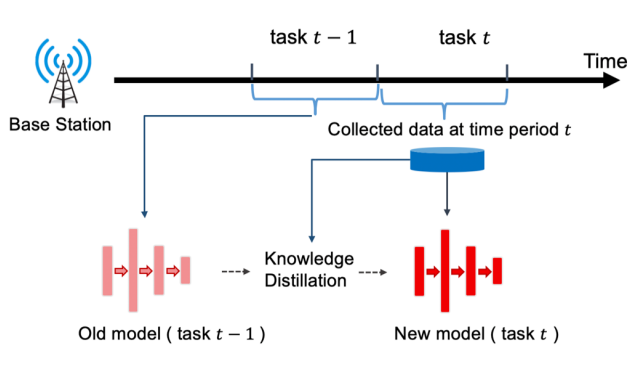


Fig. 1. The overview of CKD framework.

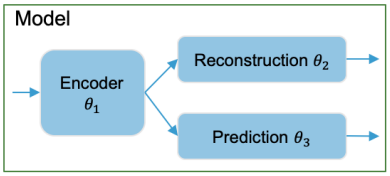


Fig. 2. The structure of model.

is assumed to preserve the knowledge of historical traffic information. When we learn a new model on the collected dataset from task t, we utilize the knowledge distillation technique to transfer the knowledge from the old model to the new model.

C. Knowledge Preservation

To learn the knowledge of historical traffic data as much as possible, we introduce a two-branch neural network model, as shown in Fig. 2. The model consists of two paths: an Encoder-Reconstruction path and an Encoder-Prediction path.

The Encoder-Reconstruction path contains the encoder module with parameter θ_1 followed by the reconstruction module with parameter θ_2. This path takes in a data point x_i and outputs $f_{rec}(x_i; \theta_1, \theta_2)$, which tries to reproduce the data x_i as similar as possible. We define the reconstruction loss as:

$$L_{rec} = \frac{1}{m} \sum_{i}^{m} d(f_{rec}(x_i; \theta_1, \theta_2), x_i), \tag{1}$$

where $d(\cdot, \cdot)$ is a distance metric between two vectors. We use $L2$-norm for d in this paper.

The Encoder-Prediction path shares the same encoder with the Encoder-Reconstruction path and then diverges into the separate prediction module with parameter θ_3. This path takes in a data point x_i and predicts the future traffic $f_{pred}(x_i; \theta_1, \theta_3)$. We define the prediction loss as:

$$L_{pred} = \frac{1}{m} \sum_{i}^{m} d(f_{pred}(x_i; \theta_1, \theta_3), y_i). \tag{2}$$

By sharing the encoder between both paths, we force extracted features by the encoder to be representative for both prediction and reconstruction purposes. In this way, the model can preserve more information of traffic data than a model

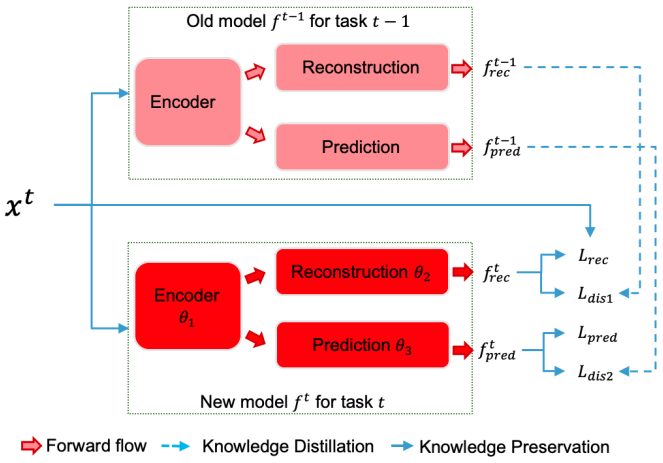


Fig. 3. The forward procedure of the model.

trained solely for prediction purposes. Although the BS cannot store all historical data, we can still leverage the knowledge preserved in the model in the future.

D. Knowledge Distillation

We adopt the knowledge distillation (KD) technique to continually update the model. Suppose we are training a new model f^t for task t ($t > 1$) and have stored the old model f^{t-1} for the task $t-1$. The parameters of f^{t-1} are fixed when updating the new model, and thus $f^{t-1}(x;\cdot)$ is simplified to $f^{t-1}(x)$. Assuming f^{t-1} contains the knowledge from historical traffic data, we propose two loss functions to distill this knowledge into f^t.

The reconstruction produced by f^{t-1} for the data point from task t includes the highly-representative knowledge of the historical data. To incorporate this knowledge into the new model for later task usage, we introduce the reconstruction distillation loss as:

$$L_{dis1} = \frac{1}{m}\sum_{i}^{m} d(f_{rec}^{t}(x_i;\theta_1,\theta_2), f_{rec}^{t-1}(x_i)). \tag{3}$$

The prediction of f^{t-1} on the data point from task t contains the traffic pattern information from historical data. We expect that the new model f^t can learn this information. To achieve this, we define the prediction distillation loss as:

$$L_{dis2} = \frac{1}{m}\sum_{i}^{m} d(f_{pred}^{t}(x_i;\theta_1,\theta_3), f_{pred}^{t-1}(x_i)). \tag{4}$$

To train a new model, we develop a joint training algorithm. This joint training algorithm is effective for both learning for the current task and incorporating the knowledge from previous tasks. To sum up, the updating of the new model is to minimize the following objective:

$$L = L_{rec} + L_{pred} + L_{dis1} + L_{dis2}. \tag{5}$$

The forward procedure of the model for task t is illustrated in Fig. 3. We first pass the data from task t through the encoder module of f^t, which extracts the hidden features. These features are fed into the reconstruction module to generate the reconstructed data as f_{rec}^t. In the meantime, these features go through the prediction module to produce the predicted traffic f_{pred}^t. Then the data pass two paths of f^{t-1} as well, resulting f_{rec}^{t-1} and f_{pred}^{t-1}. With these outputs, we can calculate the following losses. We compute the reconstruction loss L_{rec} for the new model using Eq. (1) and calculate the prediction loss L_{pred} based on Eq. (2). We also obtain two distillation losses L_{dis1} and L_{dis2} according to Eq. (3) and Eq. (4) respectively. From these losses, we construct an aggregated loss L as Eq. (5). The aggregated loss is then back-propagated to update θ_1, θ_2 and θ_3 of model f^t.

Algorithm 1: CKD Training Procedure

Input : $\{D^t\}_{t=1}^{T}$.
Output: Model f^T with parameters θ_1, θ_2 and θ_3.
1 Initialize θ_1, θ_2 and θ_3, store f^0 with θ_1, θ_2 and θ_3.
2 **for** $t \in \{1, 2, ..., T\}$ **do**
3 Obtain $x^t, y^t \in D_t$.
4 **while** *not converged* **do**
5 Pass x^t through f^t to obtain f_{rec}^t and f_{pred}^t .
6 Compute L_{rec} by Eq. (1).
7 Compute L_{pred} by Eq. (2).
8 **if** $t > 1$ **then**
9 Pass x^t through f^{t-1} to obtain f_{rec}^{t-1} and f_{pred}^{t-1} .
10 Compute L_{dis1} by Eq. (3) .
11 Compute L_{dis2} by Eq. (4) .
12 Compute L by Eq. (5) .
13 **else**
14 Compute $L = L_{rec} + L_{pred}$.
15 Update θ_1, θ_2 and θ_3 by minimizing L .
16 **end**
17 Replace f^{t-1} with f^t parameterized by θ_1, θ_2 and θ_3 .
18 **end**

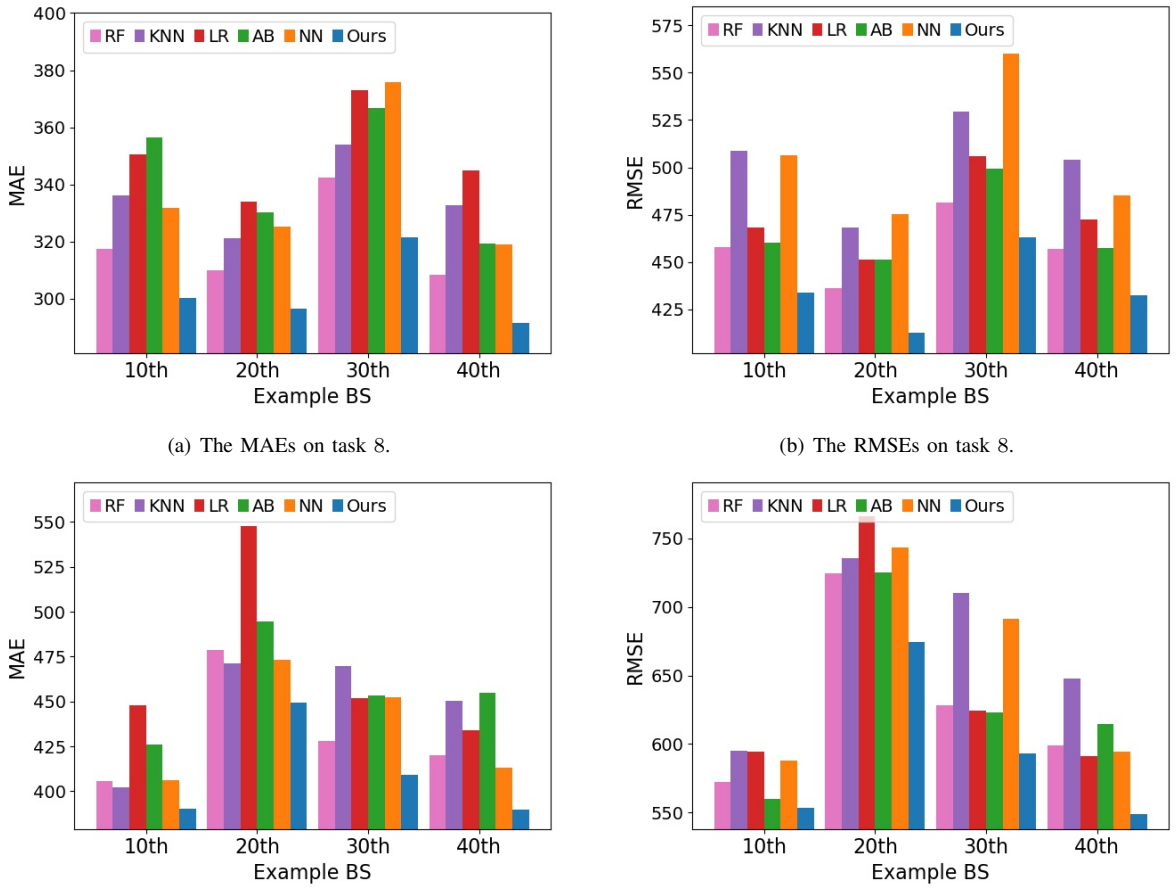
E. The Overall CKD Procedure

The whole CKD training procedure is summarized as Alg. 1. To sum up, when the first task comes, the model is updated by minimizing the reconstruction loss and prediction loss (lines 14-15). When the task number is larger than 1, the objective function to update the model includes reconstruction loss, prediction loss, and two distillation losses (lines 12,15). After training the model, we store it for later use (line 17). The testing procedure of CKD is passing the input data through the Encoder-Prediction path of the trained model, and thus we can obtain the predicted traffic.

III. Evaluation

A. Experiment Settings

1) Data Set Description: The experiments are conducted on the cellular traffic data collected from 50 base stations within an anonymous area over one year from January 1st



(a) The MAEs on task 8.

(b) The RMSEs on task 8.

(c) The MAEs on task 12.

(d) The RMSEs on task 12.

Fig. 4. The performance of different methods on the example BSs.

to December 31st. Each BS monitors traffic data every 15 minutes. Thus, the total number of data samples in each BS is $\frac{60}{15}\ samples \times 24\ hours \times 365\ days = 35,040$ [1]. We assume that each BS with the limited capacity can store **one-month** traffic data, ending up with 12 tasks. We treat task $t(1 \leq t \leq 11)$ as the training data set, and task $t+1$ as the testing data set.

2) Evaluation Metrics: This paper employs two metrics for performance evaluation as below:

Mean Absolute Error (**MAE**) calculates the average of the absolute discrepancies between the predicted traffic $\hat{y}_i$ and the actual traffic y_i, i.e.,

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|y_i - \hat{y}_i|, \tag{6}$$

where m is the number of time slots, upon which a model conducts predictions.

Root Mean Square Error (**RMSE**) represents the square root of the mean difference between the predicted traffic and the ground truth, i.e.,

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2}. \tag{7}$$

3) Training settings: The encoder module in CKD is a feed-forward neural network with four hidden layers, which has 32, 128, 512, and 128 dimensions separately. We apply the ReLU activation function after each hidden layer. The reconstruction module has two linear layers with 32 and 128 neurons respectively, and each linear layer is followed by a ReLU activation function. For the prediction module, we adopt one linear layer with 32 neurons.

In the experiments, we initialize the learning rate (lr) to 0.01 and divide (lr) by 3 if the training loss does not decrease within 3 consecutive epochs. We set the batch size to 256 and use Adam [11] as the optimizer for 50 epochs training.

B. Methods Evaluated

We implement the following baseline methods.

- Random Forest (**RF**) is an ensemble learning method that builds several decision trees during training and predicts the weighted mean result of individual trees.

[1]More detailed information of the data set cannot be released to the public, due to the agreement between us and the partner.

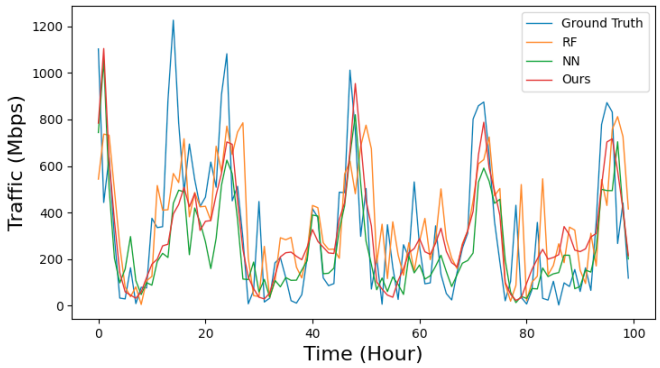


Fig. 5. The traffic prediction results on the 10th BS.

- K-nearest neighbor regression (**KNN**) predicts the value for the target object by averaging the values of the k nearest neighbors of the target object in the data set.
- Linear Regression (**LR**) models the relationship between the input factor and targets by a linear approximation.
- AdaBoost (**AB**) is a boosting learning method that adaptively learns several weak learners in favor of those instances less understood by other learners.
- Neural Network (**NN**) establishes a collection of connected neurons for the traffic prediction task. In this paper, we connect the encoder module and prediction module of CKD for building the NN structure.

C. Evaluation Results on Example Base Stations

For the sake of clarity, we first present the evaluation results on some example BSs, and the results on all BSs are to be reported in the next subsection III-D. We take the 10th, 20th, 30th, and 40th BSs as examples to demonstrate the advantage of CKD. We compare the performances of different methods on the prediction for task 8 and task 12.

Fig. 4 compares different methods in terms of MAE and RMSE, respectively. As shown in Fig. 4(a) and Fig. 4(b), CKD achieves the lowest MAE and RMSE among all approaches on task 8. For example, compared with KNN, LR, and AB on the 10th BS, the proposed CKD reduces the MAEs by up to 10.7%, 14.3%, and 15.7%, respectively. Similarly, the reductions of RMSEs by CKD are up to 14.6% compared with KNN on the 10th BS. This illustrates that, by utilizing the knowledge distillation technique, our proposed CKD framework further advances the state-of-the-art method. In addition, the results of each method on task 12 can be compared in Fig. 4(c) and Fig. 4(d). CKD outperforms RF and NN by up to 7.2% and 5.8% in terms of MAE on the 40th BS, respectively. These results suggest that CKD achieves consistent improvement over multiple tasks.

Moreover, we use 10th BS as an example to visualize the traffic prediction results. Due to the space limit, we plot the traffic prediction results of RF, NN, and our CKD in Fig. 5. It is shown that RF experiences lag in the prediction. For instance, the peak comes at around the 50th hour in the real world, while RF predicts the peak around the 53rd hour. With the help of KD, CKD achieves better prediction performances at both peak and bottom time slots.

D. Evaluation Results on All Base Stations

In this subsection, we illustrate that CKD achieves large improvements on all BSs. Table I provides the average MAE results over all BSs obtained from different tasks (i.e., task $8, 10,$ and 12). We again confirm that our proposed CKD achieves lower prediction errors than the other methods. We treat the NN as the state-of-the-art (SOTA) method. It can be observed that CKD significantly outperforms the SOTA by up to $9.62\%, 6.18\%$, and 6.74% on task $8, 10$, and 12. This confirms that our proposed knowledge distillation technique is useful in utilizing the knowledge from historical data.

TABLE I
THE AVERAGE MAE OF DIFFERENT METHODS OVER ALL BSS.

	Task		
Method	8th	10th	12th
RF	315.31	355.11	388.44
KNN	335.79	372.51	406.86
LR	344.15	380.83	418.45
AB	332.68	378.70	415.30
NN	325.23	354.22	390.87
Ours	**293.95 [-9.62%]**	**332.32 [-6.18%]**	**364.53 [-6.74%]**

TABLE II
THE AVERAGE RMSE OF DIFFERENT METHODS OVER ALL BSS.

	Task		
Method	8th	10th	12th
RF	459.08	505.97	561.62
KNN	497.79	541.17	594.05
LR	474.34	519.33	572.93
AB	460.07	510.93	566.18
NN	489.57	526.1	580.92
Ours	**432.24 [-11.71%]**	**479.65 [-8.83%]**	**526.55 [-9.36%]**

We further analyze the average RMSE results over all BSs, which also reflects the overall performance of each method. In Table II, we present the average RMSE results of each approaches that conducted on task $8, 10,$ and 12. From this table, we note that CKD outperforms all SOTA methods in terms of the average RMSE on all tasks. Compared to NN, CKD reduces the prediction RMSEs by 11.71%, 8.83%, and 9.36%, respectively. This result again confirms the advantage of our knowledge distillation-based method, comparing to the straightforward training method, such as RF, LR, and NN.

IV. RELATED WORK

A. Traffic Prediction

1) Statistical model based traffic prediction: The early traffic prediction methods are based on the statistics or probabilistic distributions of traffic data, since studies show that traffic data over a different period has certain statistical distributions. Thus, statistical models, such as mobility model [12], network traffic model [13], ARIMA [14], and α-stable model [15], have been applied into the early traffic prediction work.

However, the statistical model based methods suffer from low accuracy in reality, since the true traffic data distribution is hard to estimate, resulting in prediction errors.

2) Machine learning (ML)-based traffic prediction: Recently, ML methods, such as support vector regression [6], linear regression [16], and Long Short-Term Memory (LSTM) [17] have been employed for traffic prediction. ML-based traffic prediction estimates a function between the historical data and the predicted traffic, thus providing an improved prediction accuracy. For example, Zhang *et al.* [17] propose a convolutional LSTM-based multi-step prediction framework for modeling the temporal dependence of traffic and achieve a higher prediction accuracy than the ARIMA-based method. Zhang *et al.* [8] propose a densely connected convolutional neural networks (CNN) to learn the traffic dependence for prediction. Wang *et al.* [18] propose a hybrid deep learning based traffic prediction method by using both the autoencoder and LSTM models.

Although the above ML-based methods achieve high prediction accuracy, their accuracy relies on a large amount of data which is unavailable for the real-world BS. Unlike all the existing studies, we propose a CKD framework that utilizes the knowledge stored in old models trained on historical data to adapt the prediction model over time. Thus, we can strike a perfect balance between the prediction accuracy and the storage efficient.

B. Knowledge Distillation

Hinton et al. [19] first introduced the idea of knowledge distillation where a small student model is trained under the supervision of a large teacher model. A student model is trained not only to predict ground truths as closely as possible, but also should match soften target distributions of a teacher model. To better distill knowledge, FitNet [20] and attention transfer [21] consider the intermediate representations of teachers as additional knowledge and guide students to learn the representations. The relationships among neural layers and neurons [22], [23] are also utilized as another type of knowledge distilled by teacher models.

Overall, those knowledge distillation techniques achieve good performances on various applications. However, applying the KD on communication traffic prediction has not yet been well explored. Besides, existing KD studies are conducted on the stationary distribution. This paper develops a framework that leverages the KD to achieves high traffic prediction accuracy in a constantly changing environment.

V. Conclusion

This paper introduces CKD, a communication traffic prediction framework that can achieve high prediction accuracy under the keep changing communication scenarios. CKD embraces knowledge distillation techniques, which allows the prediction model incorporating the knowledge from historical data with the new data information. Experiments on a real-world data set illustrates the efficiency and efficacy of the proposed method.

References

[1] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5g wireless networks," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 82–89, 2014.

[2] V. Sciancalepore, K. Samdanis, X. P. Costa, D. Bega, M. Gramaglia, and A. Banchs, "Mobile traffic forecasting for maximizing 5g network slicing resource utilization," in *INFOCOM*, pp. 1–9, IEEE, 2017.

[3] I. Alawe, A. Ksentini, Y. H. Aoul, and P. Bertin, "Improving traffic forecasting for 5g core network scalability: A machine learning approach," *IEEE Network*, vol. 32, no. 6, pp. 42–49, 2018.

[4] Y. Shu, M. Yu, J. Liu, and O. W. W. Yang, "Wireless traffic modeling and prediction using seasonal ARIMA models," in *ICC*, pp. 1675–1679, IEEE, 2003.

[5] B. Liang and Z. J. Haas, "Predictive distance-based mobility management for PCS networks," in *INFOCOM*, pp. 1377–1384, IEEE Computer Society, 1999.

[6] A. Chaudhuri, "Hierarchical support vector regression for qos prediction of network traffic data," in *IML*, pp. 15:1–15:6, ACM, 2017.

[7] C. Zhang, H. Zhang, J. Qiao, D. Yuan, and M. Zhang, "Deep transfer learning for intelligent cellular traffic prediction based on cross-domain big data," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1389–1401, 2019.

[8] C. Zhang, H. Zhang, D. Yuan, and M. Zhang, "Citywide cellular traffic prediction based on densely connected convolutional neural networks," *IEEE Communications Letters*, vol. 22, no. 8, pp. 1656–1659, 2018.

[9] B. Yin, Y. Cheng, L. X. Cai, and X. Cao, "A cooperative edge computing scheme for reducing the cost of transferring big data in 5g networks," in *TrustCom/BigDataSE*, pp. 700–706, IEEE, 2019.

[10] X. Chen, Y. Li, and T. Zhang, "Reducing flash memory write traffic by exploiting a few mbs of capacitor-powered write buffer inside solid-state drives (ssds)," *IEEE Trans. Computers*, vol. 68, no. 3, pp. 426–439, 2019.

[11] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR (Poster)*, 2015.

[12] F. Ashtiani, J. A. Salehi, and M. R. Aref, "Mobility modeling and analytical solution for spatial traffic distribution in wireless multimedia networks," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 10, pp. 1699–1709, 2003.

[13] K. Tutschku and P. Tran-Gia, "Spatial traffic estimation and characterization for mobile communication network design," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 5, pp. 804–811, 1998.

[14] H. Zare Moayedi and M. A. Masnadi-Shirazi, "ARIMA model for network traffic prediction and anomaly detection," in *2008 International Symposium on Information Technology*, pp. 1–6, 2008.

[15] R. Li, Z. Zhao, J. Zheng, C. Mei, Y. Cai, and H. Zhang, "The learning and prediction of application-level traffic data in cellular networks," *IEEE Trans. Wireless Communications*, vol. 16, no. 6, pp. 3899–3912, 2017.

[16] H. Sun, H. X. Liu, H. Xiao, R. R. He, and B. Ran, "Use of local linear regression model for short-term traffic forecasting," *Transportation Research Record*, vol. 1836, no. 1, pp. 143–150, 2003.

[17] C. Zhang and P. Patras, "Long-term mobile traffic forecasting using deep spatio-temporal neural networks," in *MobiHoc*, pp. 231–240, ACM, 2018.

[18] J. Wang, J. Tang, Z. Xu, Y. Wang, G. Xue, X. Zhang, and D. Yang, "Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach," in *INFOCOM*, pp. 1–9, IEEE, 2017.

[19] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, 2015.

[20] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," *arXiv preprint arXiv:1412.6550*, 2014.

[21] N. Komodakis and S. Zagoruyko, "Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer," in *ICLR*, 2017.

[22] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *CVPR*, pp. 7130–7138, IEEE Computer Society, 2017.

[23] S. Lee and B. C. Song, "Graph-based knowledge distillation by multi-head attention network," in *BMVC*, p. 141, BMVA Press, 2019.