Self Optimizing Network Slicing in 5G for Slice Isolation and High Availability

Shwetha Vittal and Antony Franklin A
Department of Computer Science and Engineering, Indian Institute of Technology Hyderabad
Email: {cs19resch01001, antony.franklin}@iith.ac.in

***Abstract*—5G network supports end-to-end logically isolated networks in the form of network slices, catering to the needs of users of various primary network services, namely enhanced Mobile Broadband (eMBB), ultra Reliable Low Latency Communications (uRLLC), and massive Machine Type Communication (mMTC). Mobile Virtual Network Operators (MVNO)s often face challenges in achieving strong slice isolation and High Availability per slice during overload and scaling situations as the 5G network uses a shared environment for slices with multiple domains, especially considering a variety of services and devices. In this paper, we propose a novel Self Optimizing Network Slicing framework (SONS) leveraging Self Organizing Network by building it as an autonomous slice system in 5G network slicing management for efficient slice sharing and isolation. Precisely, we formulate a system model with Probabilistic Graphical Model (PGM) based Markov Network, building it as an Artificial Intelligence based learning framework. We propose Slice Belief Propagation based algorithms and Deep Learning based Long Short Term Memory (LSTM) methods to aid in serving user requests and reconfiguration of self optimizing slice. Our experiments on the proposed SONS framework shows improvement in serving higher number of users with uninterrupted connectivity by 80% in eMBB, 35% in uRLLC, and 52% in mMTC when compared to standard slice deployments, while handling the worst case of peak traffic in the control plane of 5G Core network.**

I. Introduction

Network slicing [1] has emerged as a strong enabler in 5G system leveraging Network Function Virtualization (NFV) and Software Defined Networking (SDN) to meet the diverse market segments' requirements in enhanced Mobile Broad Band (eMBB), ultra Reliable Low Latency Communication (uRLLC), and massive Machine Type Communication (mMTC). It uses a common physical infrastructure with virtualised resources in terms of storage, computation, and networking across multiple domains to ensure the smooth and secure functioning.

In this context, we envisage to build an optimized network slicing to smoothen the 5G network performance by making the network slice more dynamic and adaptable to varying traffic conditions and improve the user experience with High Availability (HA), which otherwise are too complex to be configured manually. We propose a novel Self Optimizing Network Slicing (SONS) framework in the 5G Core Network (5GC) by stitching Self Organizing Network (SON) with Deep Learning (DL) techniques from Artificial Intelligence (AI) in a distributed and Cross Slice Communication (CSC) environment to optimize the operation of network slice in terms of the resources required to dynamically re-configure it and handling the arrived User Service Request (USR)s in control plane by efficient utilization of these resources.

II. Modes of Network Slice Functioning and Problem Description

A. Background

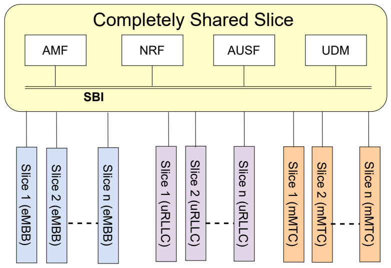
Functioning of 5GC is mainly driven by the control plane NFs working together to serve different USRs which require access to various slice services in eMBB, uRLLC, and mMTC categories. Access and Mobility Management Function (AMF), Authentication Server Function (AUSF), Unified Data Management (UDM), Network Repository Function (NRF), and Session Management Function (SMF) are some of the primary control plane NFs [2] which can be grouped together to compose a slice, to provide control plane services to the UEs, like UE registration, Packet Data Unit (PDU) session establishment and modification, and UE deregistration. Typically Network Slice Management Function (NSMF) [3] deploys a slice to function either in shared mode or completely isolated mode using one or more network slice subnet(s) and associated NF(s).

B. Slice in Completely Shared Mode

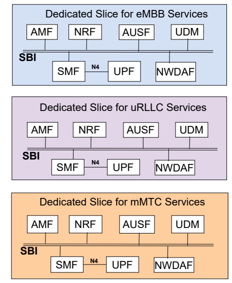
In the completely shared mode of slice functioning, all the services provided by each of the NFs of the slice instance are common and shared among other slice instances which rely on it for the respective slice service. Fig. 1a shows the completely shared slice composed of 5GC control plane primary NFs, namely AMF, NRF, AUSF, and UDM shared by each of the specific slice instance made of User Plane Function (UPF) and SMF NFs for provisioning the actual slice type service.

C. Slice in Completely Isolated Mode

In the completely isolated mode, specific NFs are created and deployed to be part of the required slice instance to provide the respective services from the slice. Here, slices neither share the resources between them nor communicate with each other. Fig. 1b shows the completely isolated slices for each slice service category, composed of both control plane and data plane NFs, namely AMF, NRF, AUSF, UDM, Network Data Analytics Function (NWDAF), SMF, and UPF in 5GC.



(a) Completely shared mode of slice functioning.



(b) Completely isolated mode of slice functioning.

Fig. 1: Different modes of slice functioning in 5GC.

D. Problem Description

The shared slice environment is very helpful in saving the resources when multiple slices provide the similar functionalities like the 5GC control plane. But, an unforeseen situation of congestion, overload, scaling, or fault recovery happening in the system, makes the shared slice unavailable and hence drops USRs arriving during this situation. This would undesirably affect the SLA of all those slices which rely on the shared slice. Therefore, strong slice isolation is a must to operate parallel slices. Dynamic traffic conditions in 5GC, urges MVNOs to dynamically scale out/in the respective slice with respect to its NFs for efficient resource utilization. However, getting a slice into an operational state in order to serve the respective USRs, consumes quite an amount of time (about 45 seconds as observed in one of our previous works [4]) in NFV, as it involves instantiation, configuration, and activation of the related NF(s) of slice. Hence, even in the completely isolated mode of slice, the USRs get dropped during these conditions resulting in an unpleasant environment. Thus, this trade-off between completely shared and completely isolated functioning of slice(s) opens up challenges to MVNO(s) to handle the overall complexity aroused to meet the High Availability (HA) service demands of users.

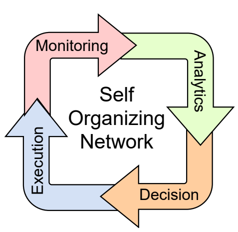


Fig. 2: Self Organizing Network.

III. Related Work and Motivation

In the literature, sincere efforts are on, to achieve smooth functioning of slice operation across end-to-end domains of the slice. Authors in [5, 6], highlighted the challenges when parallel slices are being operated and the trade-off introduced by network slicing between customized slicing and dynamic resource sharing on a common shared underlying substrate. Authors in [7] highlighted the need of CSC for services deployed using slices co-located in edge cloud infrastructures and hence, Inter Slice Communication (or CSC [8]) is yet to be embraced.

In close relation to the cognitivity, SON [9] concepts introduced in 4G can be reused for 5G to improve the operational sustainability. SON (depicted in Fig. 2) definitely motivates the need for AI and Big Data Analytics as a crucial enabler to move from automation to autonomic system in 5G. Self-optimization being part of SON functions, moves away from manual intervention by the operator to aim at maintaining network performance by regularly monitoring and analyzing the performance data of the network in a Closed Loop Automation.

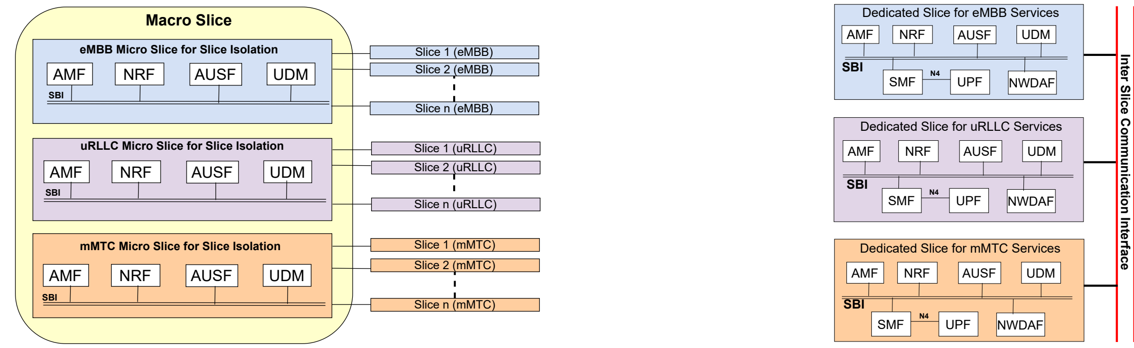
Therefore, the motivation for our work is two directional.

- One side is on handling and serving of USRs efficiently by focusing on achieving the slice isolation in 5GC framework.
- Other side is to achieve the HA with respect to the operation of the slice in 5GC by leveraging CSC with efficient utilization of the available resources to the slice using SON and distributed deep learning mechanisms from AI.

To the best of our knowledge, this work is the first attempt to show that the operation of slice functioning in the control plane of 5GC can be improved to meet the requirements and demands of HA by leveraging the CSC and SON along with AI, involving all the primary NFs of the respective slice, without compromising on slice isolation. Hence, we propose a tightly integrated framework of network slice sharing and isolation to achieve self optimization in the operation of network slices in 5GC.

To summarize, the key contributions of this work are:

- A framework to facilitate the self optimization of network slices in the 5GC.
- System model for self optimization of the network slicing framework to enable learning and cognitivity in the operation of slices.
- Algorithms for self optimization which make efficient usage of underlying resources in serving the USRs and slice reconfiguration.



(a) Self optimized slice functioning in intra-slice mode.

(b) Self optimized slice functioning in inter-slice mode.

Fig. 3: Self optimized modes of slice functioning in 5GC.

IV. Self Optimizing Network Slicing Framework

A. Proposed Self Optimizing Mode of Slice

In the proposed self optimizing mode, the slices function similar to completely isolated mode. But, the slices assist each other during situations of dynamic traffic conditions like slice overload, congestion, and failure recovery, to avoid degradation of services, thereby optimize themselves in functioning. It is most relevant to situations when a single MVNO deploys multiple slices of same or different categories to cater a specific SLA. Precisely, slice in SONS mode operates

- To serve the USRs of other slices in needed situations.
- To get assistance from other slices in SONS mode to serve its own USRs.

Here, we define two sub categories of functioning of SONS mode, a) intra-slice SONS mode and the b) inter-slice SONS mode.

1) Intra-Slice SONS Mode: Quite a set of NFs can be availed by MVNO composing a macro slice to cater different slice service demands of users. Further this macro slice requirement can be tailored and hence sliced to satisfy a specific vertical and service demands of a use case in the form of micro slice. In the intra-slice SONS mode there is a single macro slice instance with multiple micro slice instances within it. Each micro slice instance is dedicated to serve the respective USRs of the slice service category (eMBB, uRLLC, and mMTC). However, single macro slice as a whole is shared. So it uses the NFs from the available micro slice (e.g: least loaded micro slice) to serve the USRs of micro slice in need. Fig. 3a shows the intra-slice SONS functioning in 5GC with AMF, AUSF, NRF, and UDM NFs.
2) Inter-Slice SONS Mode: In this SONS mode there is a specific slice instance for each of the eMBB, uRLLC, and mMTC slice categories with AMF, NRF, AUSF, and UDM along with SMF and UPF NFs similar to completely isolated method but the slices communicate with each other to aid each other as per the need in SONS mode. Fig. 3b shows the inter-slice functioning of SONS in 5GC.

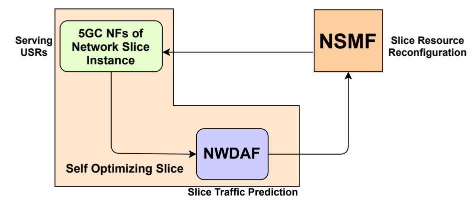


Fig. 4: Proposed self optimizing network slicing framework.

B. Proposed SONS Framework

In the proposed SONS framework, various measurements are performed by closely monitoring Key Performance Indicator (KPI)s of the slice (in terms of data volume, the number of registered UEs, the number of PDU sessions, etc.) while serving the USRs at different NFs of the slice in 5GC for analysis at NWDAF.

Fig. 4 shows the proposed SONS framework facilitating the self optimization functioning of the involved network slice with three important components, namely:

- NSMF: It is responsible for the deployment of SONS and re-configure the SONS with related NFs' resources (like number of instances of each NF) for its optimized functioning.
- SONS instance: Actual SONS instance with related 5GC NFs to serve the USRs (Fig. 3a, Fig. 3b).
- NWDAF: It performs slice analytics and feeds it's output to NSMF to aid in slice's own reconfiguration in the next reconfiguration period.

V. System Model of the Proposed SONS Framework

To facilitate the working of SONS, we consider a set of active network slices forming an undirected Markov Network using Probabilistic Graphical Model (PGM) [10]. Unlike Bayesian networks, Markov Network model allows great flexibility in representing interactions between variables with symmetricity without directional influence and without any specific order of dependency. Algorithms 1, 2, 3, and 4 enable the working of SONS (see Fig. 4) for slice reconfiguration and serving the USRs in the control plane.

A. Representation

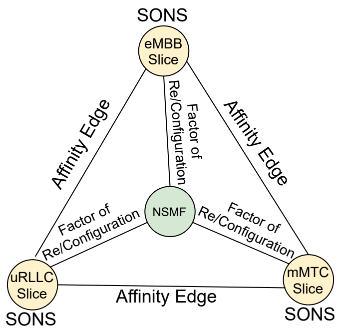


Fig. 5: Probabilistic Graphical Model with Markov Network of self optimizing network slices.

We build the complete Markov Network with a set of nodes and edges. As shown in Fig. 5, let G be an undirected graph based Markov Network. Each node or vertex here showcases either a self optimizing slice or the NSMF. Edge between SONS nodes indicates a factor of affinity, capturing the strength of influence or the interaction between the respective slices. Edge between NSMF and SONS node indicates the strength of the available capacity of re/configured resources in the form of NFs for SONS by NSMF. Let 'n' be the number of SONS deployed by NSMF to serve the traffic for their respective services. Markov Blanket [10, 11] is defined for

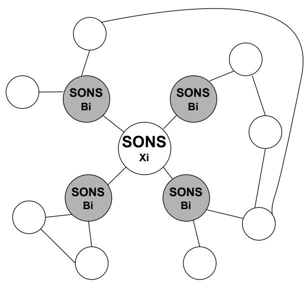


Fig. 6: Markov blanket of SONS with Markov Network of self optimizing network slices.

each slice in the system. It identifies and covers the neighbour slices for the respective slice by defining a local independence for it with respect to other slices. Let B_i be the Markov Blanket defined for slice i in Markov Network G. As shown in Fig. 6, the Markov Blanket B_i of SONS node X_i is a subset of SONS set $S = \{X_1, X_2, \ldots, X_n\}$ conditioned on other SONS which are independent with respect to X_i as

$$X_i \perp\!\!\!\perp S \backslash B_i | B_i \tag{1}$$

B. Learning

The learning per SONS occurs in distributed mode of SONS framework at different NFs of Markov Network of slices and final output is fed to NSMF in a closed loop manner, thereby harmonizing the distributed deep learning of AI with PGM.

Algorithm 1: Dynamic reconfiguration of self optimizing slice.

Input: $availableSlices$ at time i
for $sliceIndex \leftarrow 1$ **to** $numberOfAvailableSlices$ **by** 1 **do**
  $predictedResourceInfo$ $\leftarrow$ NWDAF.getPredictedServingUSRInfo ($sliceIndex$);
  updatePredictedInfoOfSlice($sliceIndex$);
for $sliceIndex \leftarrow 1$ **to** $numberOfAvailableSlices$ **by** 1 **do**
  $sliceContext \leftarrow$ getSliceInfo(sliceIndex);
  $resourceToAllocateInfo$ $\leftarrow$ computeResourcesToAllocate ($sliceContext, sliceContext.neighbourSlices$);
  reconfigureSlice($resourceToAllocateInfo$);

Algorithm 2: Slice Belief Propagation among self optimizing slices in SONS framework.

Input: Markov Blanket(B_i) at time i
for $sliceIndex \leftarrow 1$ **to** $numberOfSlices$ in B_i **by** 1 **do**
  $neighborSlice$ $\leftarrow$ getNeighborSliceContext($sliceIndex$);
  $neighborSliceInfo$ $\leftarrow$ getBeliefFromNeighborSlice($neighborSlice$);
  updateNeighborSliceBelief ($neighborSliceInfo$, $neighborSlice$);

1) Learning for reconfiguration of the SONS
2) Learning by SONS for serving USRs

Algorithm 3: Serving user request with self optimizing macro slice.

Input: Incoming User Request(R_i) at time i
$requestSliceType \leftarrow$ getRequestSliceType(R_i);
$sliceInstance \leftarrow$ getMatchingMicroSliceInstance ($requestSliceType$);
if !$sliceInstance$ **then**
  $sliceInstance \leftarrow$ getServingMicroSliceInstance ($requestSliceType$);
if $sliceInstance$ **then**
  $sliceInstance$.serveUSR(R_i);
  NWDAF.LSTM.feedRequestHandledInput ($requestSliceType$,R_i);
else
  handleDroppedUSR(R_i);

1) Learning for Reconfiguration of the SONS: This learning focuses on predicting the USRs per SONS to help in its reconfiguration. Considering the effectiveness of Recurrent

Algorithm 4: Serving user requests with self optimizing slice in inter-slice functioning mode.

```
Input: Incoming User Request(R_i) at time i
requestSliceType ← getRequestSliceType(R_i);
requestHandled ← false;
if sliceAvailable then
    if requestSliceType == selfSliceType then
        serveUSR(R_i);
        requestHandled ← true;
    else
        requestedServingSlice
          ← getRequestedServingSliceInfo
                              (requestSliceType));
        if requestedServingSlice then
            requestedServingSlice.serveUSR(R_i);
            requestHandled ← true;
else
    sliceInstance ←
      getSBPLikelyServingSliceInstance(R_i);
    redirectUSRToServingSlice(sliceInstance,R_i);
if requestHandled then
    NWDAF.LSTM.feedRequestHandledInput
                          (requestSliceType,R_i);
    return success
else
    handleDroppedUSR(R_i);
    return failure
```

Neural Networks (RNN) from DL techniques of AI, we use Long Short Term Memory (LSTM) [12] to model the non-linear co-relation between the past and the current data points with respect to incoming USRs per slice. We place this LSTM model as a predicting engine running at NWDAF per slice. It takes USRs arrived and honoured by the other NFs of the slice like AMF as inputs.

The LSTM model is trained during each time window, based on the incoming USRs at the slice for all the previous time windows. It then forecasts the number of USRs arriving in the next time window. The output from this model is used as an input to the NSMF. Based on the periodicity of slice reconfiguration i.e., considering at least 30 minutes to 1 hour as per [6], NSMF invokes the proposed SONS reconfiguration shown in Algorithm 1. At first, it updates the predicted USRs output from the slice's NWDAF for all the available slices. It then recomputes the resources for each such slice (in terms of the number of instances of every NF, related storage, and compute involved in the slice).

2) Learning by SONS for serving USRs: This learning focuses on handling the USRs of the slice arriving at NFs, namely AMF and SMF. The USRs here register and request service on the respective slice. The learning agent at the NFs uses the monitored KPIs on the slice in the form of number of active users, number of past registered users, current CPU and memory load in handling various USRs, and number of PDU sessions in a similar way. The agent shares this information with neighbouring slices identified by Markov Blanket. These KPIs form the learning parameters to self as well as to the neighbouring slices, thereby helping in getting local factors of strength of interaction between all the neighbouring slices in the scope of Markov Blanket of the given Markov Network.

C. Inference

The proposed inference method is motivated from Belief Propagation (BP) and Loopy Belief Propagation (LBP) mechanisms [13]. Here, we consider inferring the most likely influencing slice using Slice Belief Propagation (SBP) inference method.

Slice Belief Propagation: In this proposed method, we consider interaction between SONS in a pairwise manner. Given a pair of neighboring SONS nodes, there is only one pairwise interaction but messages flow in both directions. The learning and inference by SBP shown in Algorithm 2 runs at every SONS based on a periodicity. As shown in this algorithm, given the set of neighbouring slices from the input Markov Blanket, the SONS fetches the neighbouring slice information as the learnable parameters (described in Section V-B2) and updates this information in the respective slice context for deriving the most probable influencing slice to assist it.

The proposed SONS algorithms for intra- and inter-slice functioning shown in Algorithm 3 and Algorithm 4 serve the USRs by first checking the availability of matching serving slice. If the serving slice is not available, it redirects the request to the most likely slice to serve the USR. Upon successful USR processing, related information is fed to NWDAF for additional learning (Section V-B1) towards SONS' reconfiguration.

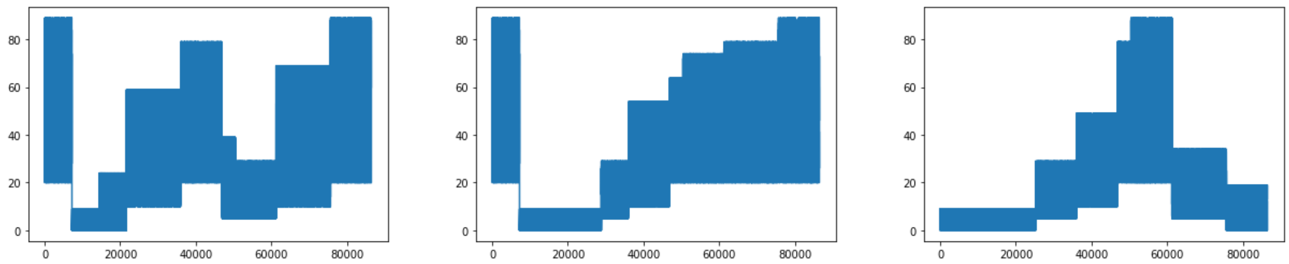
VI. Performance Evaluation

A. Experimental Setup

To evaluate the performance of the proposed SONS algorithms (detailed in Section V) and compare it with other two modes (detailed in Section II), we developed a simulation framework individually for all the three different slice functioning modes using C++. Specifically for SONS mode, we modelled the Markov Network as per pgmpy [14], a python library for working with PGM. While 5GC control plane USRs are fed to the system continuously using an input traffic profile file, we consider an unhealthy condition with slice being overloaded and scaled out in all the three modes, to capture the related metrics of honoring these USRs.

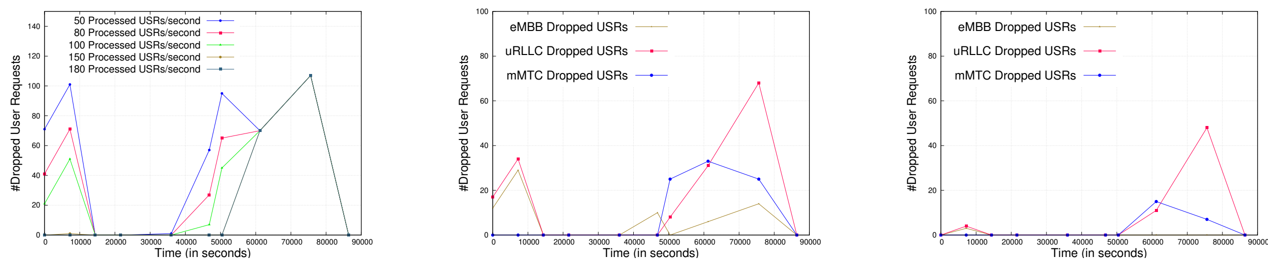
B. Performance of Learning by SONS in serving USRs

As it is difficult to get the real data set for all the generic slice types from the service providers, we used the Randomized Traffic Generator simulation model for generating the USRs load pattern similar to [15], by using an input traffic profile. However, we have considered the incoming requests



(a) (b) (c)

Fig. 7: User requests for a) eMBB slice, b) uRLLC slice, and c) mMTC slice in 24 hours with granularity of 1 second.



(a) (b) (c)

Fig. 8: Dropped user requests in a) Completely shared slice, b) Completely isolated slice, and c) Self optimizing slice.

pattern for eMBB as per [16], uRLLC as per [17], and mMTC as per [18], to generate the number of USRs (maximum 100) connecting to 5GC per second for one complete day. As per these referred works for generation of data, the traffic in eMBB and uRLLC slice types are generally peak in the night from 9 *PM* to midnight and reduces to very low utilization in the early morning hours before slowly starting to raise in the day time. However in mMTC slice type, the traffic is observed to be peak in working hours of the day only. Accordingly, Fig. 7 shows the generated USRs accessing the network in respective slice types (Fig. 7a, Fig. 7b, and Fig. 7c).

Fig. 8 shows the dropped USRs in the completely shared mode [Fig. 8a], completely isolated mode [Fig. 8b], and SONS mode [Fig. 8c] of slice functioning during unhealthy condition, in the current time window. We have assumed the scaling to be happening fairly in all the three modes with ground truth capacity of processing 20 USRs per second as per [19]. As observed here, the SONS mode drops the least amount of USRs compared to other two modes. While SONS and completely isolated modes of slice functioning are able to distinguish the USRs based on the requested respective slice types, completely shared mode does not do so, as it completely shares the services provided by the related NFs.

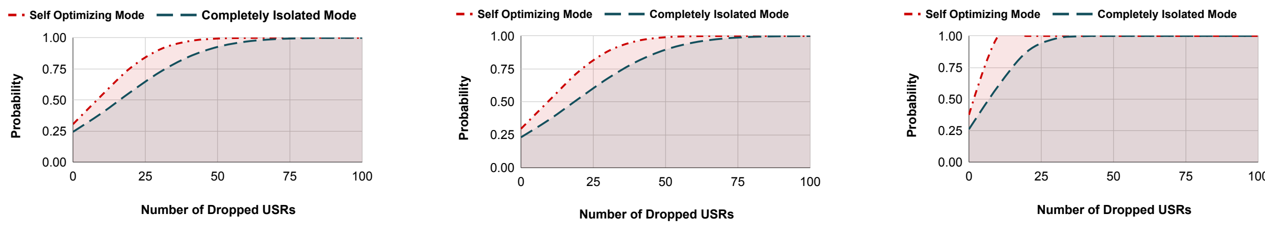
Fig. 9 shows the Cumulative Distribution Function (CDF) of the number of dropped USRs for SONS and completely isolated modes for eMBB in Fig. 9a, uRLLC in Fig. 9b, and mMTC in Fig. 9c. We can confirm that the proposed SONS mode of slice functioning shows high probability (highest by mMTC due to peak traffic only during working hours of the day) of dropping less number of USRs across all the slice types compared to completely isolated mode.

C. Performance of Learning for Reconfiguration of SONS

To evaluate the performance of this learning based on LSTM model, we used the real data set from [20]. The data set is from a bike sharing system accessing cellular network dynamically. We map this system to uRLLC slice on 5GC for evaluation. Here, we consider the hourly basis arrival of different users to align with the dynamic reconfiguration (see Section V-B1). We design LSTM using the Keras [21] APIs built on top of TensorFlow 2.0 [22] library with original dataset being split into two separate data sets. The first 65% of the data is used for training the LSTM using 64 to 128 batch sizes up to 100 epochs. Remaining 35% are set aside to test the robustness. Fig. 10a compares the predicted USRs with 65% trained and 35% tested data against the actual USRs accessing the network, in periods of hours. Fig. 10b shows the loss with prediction against the trained data by LSTM. These results confirm the efficient working of LSTM and hence serves as an useful input to the NSMF for the reconfiguration of SONS. Further enhancements can be done by tuning the associated parameters in the experiment to analyze the errors in other cases like under fitting or over fitting.

VII. Conclusion and Future Work

This paper proposed a Self Optimizing Network Slicing framework in 5G Core with related algorithms using Probabilistic Graphical Model by building Markov Network of self optimizing network slices for performing learning in distributed mode at various levels in 5GC. Our study with experiments has shown that the proposed SONS mechanism ensures slice isolation service in the normal situation and cooperates with other slices when needed to serve incoming



(a) (b) (c)

Fig. 9: CDFs of number of dropped user service requests in a) eMBB slice, b) uRLLC slice, and c) mMTC slice.

(a) Trained and predicted user versus actual requests to slice.

(b) Loss with traffic forecasting in self optimizing slice.

Fig. 10: Performance of learning for reconfiguration of SONS.

USRs by maximizing the available resource utilization and minimizing the dropping of USRs. Hence, it can be adapted flexibly in isolated mode and shared mode of functioning to serve USRs and provide HA, compared to existing completely shared and completely isolated modes. In future, we plan to extend the working of SONS for data plane of the 5G Core slice.

Acknowledgment

This work has been supported by the Department of Telecommunications, Ministry of Communications, India as part of the “Indigenous 5G Test Bed” project.

References

[1] A. Kaloxylos, “A Survey and an Analysis of Network Slicing in 5G Networks”, *IEEE Communications Standards Magazine*, vol. 2, no. 1, pp. 60–65, 2018.

[2] 3GPP, “System Architecture for the 5G System”, Tech. Rep. TS 23.501, 2018.

[3] 3GPP, “Study on Management and Orchestration of Network Slicing for Next Generation Network”, Tech. Rep. TS 28.801, 2018.

[4] S. Vittal, M. K. Singh, and A. Antony Franklin, “Adaptive network slicing with multi-site deployment in 5g core networks”, in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, 2020, pp. 227–231.

[5] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, “Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges”, *IEEE Communications Magazine*, vol. 55, no. 5, pp. 80–87, 2017.

[6] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Pérez, “Resource Sharing Efficiency in Network Slicing”, *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 909–923, 2019.

[7] I. Fotoglou, G. Papathanail, A. Pentelas, P. Papadimitriou, V. Theodorou, D. Dechouniotis, and S. Papavassiliou, “Towards cross-slice communication for enhanced service delivery at the network edge”, in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, 2020, pp. 22–28.

[8] ITU-T, “ Machine learning in future networks including IMT-2020: Use cases”, Tech. Rep. ITU-T Y.3170-Series Supplement 55, 2019.

[9] Olav Osterbo and Ole Grondalen, “Benefits of self-organizing networks (son) for mobile operators”, *Journal of Computer Networks and Communications*, vol. 54, no. 7, 2012.

[10] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques - Google Books*.

[11] “Markov Blanket”, https://www.sciencedirect.com/topics/computer-science/markov-blanket.

[12] R. Fu, Z. Zhang, and L. Li, “Using LSTM and GRU neural network methods for traffic flow prediction”, in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, 2016, pp. 324–328.

[13] Alexander T. Ihler, John W. Fischer III, and Alan S. Willsky, “Loopy Belief Propagation: Convergence and Effects of Message Errors”, *J. Mach. Learn. Res.*, vol. 6, pp. 905–936, December 2005.

[14] Ankur Ankan and Abinash Panda, “pgmpy: Probabilistic graphical models using python”, 01 2015, pp. 6–11.

[15] Mohit Kumar Singh, Shwetha Vittal, and A Antony Franklin, “Serens: Self regulating network slicing in 5g for efficient resource utilization”, in *2020 IEEE 3rd 5G World Forum (5GWF)*, 2020, pp. 590–595.

[16] Z. Luo, C. Wu, Z. Li, and W. Zhou, “Scaling Geo-Distributed Network Function Chains: A Prediction and Learning Framework”, *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 8, pp. 1838–1850, 2019.

[17] Tejas Subramanya, Davit Harutyunyan, and Roberto Riggio, “Machine learning-driven service function chain placement and scaling in MEC-enabled 5G networks”, *Computer Networks*, vol. 166, pp. 106980, 2020.

[18] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang, “Large-Scale Measurement and Characterization of Cellular Machine-to-Machine Traffic”, *IEEE/ACM Transactions on Networking*, vol. 21, no. 6, pp. 1960–1973, 2013.

[19] I. Alawe, A. Ksentini, Y. Hadjadj-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.

[20] Hadi Fanaee-T and Joao Gama, “Event labeling combining ensemble detectors and background knowledge”, *Progress in Artificial Intelligence*, pp. 1–15, 2013.

[21] “Keras: The Python Deep Learning API”, https://keras.io/.

[22] “Tensorflow: An Open-Source Machine Learning Framework for Everyone”, https://www.tensorflow.org/.