Enhanced Decision Mechanism for RAN Subslicing in Management Closed Control Loop

Marika Kulmar
Thomas Johann Seebeck Department of Electronics
Tallinn University of Technology
Tallinn, Estonia
marika.kulmar@taltech.ee

Ivo Müürsepp
Thomas Johann Seebeck Department of Electronics
Tallinn University of Technology
Tallinn, Estonia
ivo.muursepp@taltech.ee

Muhammad Mahtab Alam
Thomas Johann Seebeck Department of Electronics
Tallinn University of Technology
Tallinn, Estonia
muhammad.alam@taltech.ee

***Abstract*—5G radio access network (RAN) slicing enables better satisfaction of different quality of service (QoS) requirements than without network slicing. A handful of slice types are specified for various service verticals; however, the number and size of those slices is unspecified. Subslicing refers to grouping slice users into smaller groups, subpartitioning slice bandwidth, and allocating smaller bandwidth parts to smaller user groups. State of the art subslicing has been done to better satisfy the QoS requirements inside the service vertical. Slice performance improvement was not the purpose of subslicing, but the positive effect was noticeable. In this paper, the subslicing decision is done with the aim of improving slice performance. The decision mechanism for management closed control loop is proposed. The input dataset consists of 6 key performance indicators, namely slice bandwidth utilization, slice goodput (application level throughput) per one allocated resource block (RB), and slice block error ratio (BLER), both in uplink and downlink. This dataset is clustered, and the result is learned by a classifier to decide whether the slice is too large and should be split or too small and should be merged with another too small slice or subslice. The results show that by knowing the slice utilization and goodput per one allocated RB, the slice reconfiguration action regarding subslicing can be determined using machine learning tools.**

***Index Terms*—subslicing, performance, clustering, neural network**

I. Introduction

Network slicing in a 5G radio access network (RAN) guarantees service-level agreement (SLA) using logical networks on the physical infrastructure. These logical networks, called slices, can be configured to satisfy specific quality of service requirements and connectivity. There are no limitations on the number or size of slices set by standardization.

The slice provisioning solutions allocate sufficient resources to user equipments (UEs) admitted to the slice. In RAN, the radio spectrum resource is available in bandwidth parts (BWP) of a fixed size [1], but can be allocated to the slice with the granularity of one resource block (RB). The RB is defined in timescale as a slot time and in frequency scale as 12 subcarriers. The length of a slot time and width of a subcarrier depend on subcarrier spacing configuration [2].

Given complete slice performance data, that is, the slice performance at any slice size, the dependence of slice performance on slice size can be determined. Subsequently, slices that are too large can be split into subslices, and slices or subslices that are too small can be merged. This decision mechanism can be used in ”Decide” step of management closed control loop specified in 3GPP TS 28.535 [3].

How can we evaluate which subslice is too small and which is too large? How can we determine which subslice can be split and which one can be merged with another subslice that is too small? In this paper, it is proposed the clustering of subslice performance data into three clusters to determine whether the subslice should be merged, split, or not changed. Subsequently, the result of the performance data clustering is learned as a classification to decide suitable actions to reconfigure slices and subslices to improve the performance of a slice and network. Subslicing enables to serve more UEs if no additional bandwidth is available or if carrier aggregation is not possible.

The remainder of this paper is organized as follows. In Section II, the related work on slice and subslice size described. The proposed slice performance data clustering algorithm is described in Section III. Section IV contains the learning of clustering result as a classification. Finally, Section VI concludes the paper.

II. Related Work

In RAN slicing, optimized resource allocation determines the slice size. Optimized resource allocation to a slice is expected to result in the best slice performance. In [4], resources were allocated to subslices based on optimization, and the channel bandwidth remained constant. Their results showed that their proposed algorithm for resource allocation to subslices outperformed the others multiple times if the number of subslices was higher. However, the UE SLAs varied; thus, the effect of the number of subslices on the cell performance was not comparable. It is not known if the number of UEs

Author’s version of the paper accepted in 2023 Eighth International Conference on Fog and Mobile Edge Computing (FMEC). DOI: 10.1109/FMEC59375.2023.10306223 ©2023 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works

and average SLA for a slice were set equal among all number of subslices the slice was divided into.

The subslicing presented in [5] includes UE features selected using a support vector machine (SVM) and UEs clustered by selected features using k-means. The number of subslices was determined by evaluating the clustering quality, and performance was not considered when creating subslices. UEs with more similar requirements should work in one subslice; however, if the number of UEs is low, then the subslice is small and can exhibit poor performance.

Second, it is questionable to measure the effect of subslicing on performance change if the planned use of capacity does not match the allocated resources properly, that is, using requested data rates that are not capable of consuming the given BWP, or if the sum rate of UEs is too variable in time to evaluate the slice performance at different slice sizes.

If there are many UEs in the slice, the slice BWP is large, and if there are few UEs in the slice, the slice BWP is small. However, BWPs that are too small cannot provide the same SLA satisfaction for the UE as if the UE is admitted to the slice that uses a large BWP. The performance of a small subslice can be reduced, as discussed in [6]; if the BWP is smaller than 20 physical RBs, then the different control blocks can puncture each other and increase the coupling loss in 5G NR.

III. Performance Data Clustering

The aim of clustering the subslice performance data is to determine the action for subslice modification. The possible subslice modification options are “merge”, “no change” and “split”, thus three clusters are required. The execution of subslice modification is beyond the scope of this paper.

A. Subslice performance data

The subslice performance data were collected from the simulation results in the MATLAB 5G Toolbox system-level simulation tool called NR Cell Performance Evaluation with Physical Layer Integration [7]. The range of subslice size is 4-275 RBs (720 kHz—49.5 MHz, using subcarrier spacing of 15 kHz). Each RB in a subslice is expected to be consumed by one UE, which requests rates of 500 kbps in the UL and 667 kbps in the DL. The requested rate is served using packet sizes of 1500 bytes, 500, 150, 50, and 40 bytes. Each UE is located up to a distance of 173 m from the gNB and is expected to have good signal coverage. The key performance indicators (KPI) are subslice bandwidth utilization in UL (utilUL) and DL (utilDL), subslice goodput (application-level throughput) per one RB in UL (gdp1UL) and DL (gdp1DL), and the subslice average block error ratio (BLER) in UL (blerUL) and DL (blerDL). Further information regarding the dataset is provided in [8], section 2. Subslice performance improvement means that subslice utilization decreases, while subslice goodput increases. The subslice performance data is shown in Fig. 1

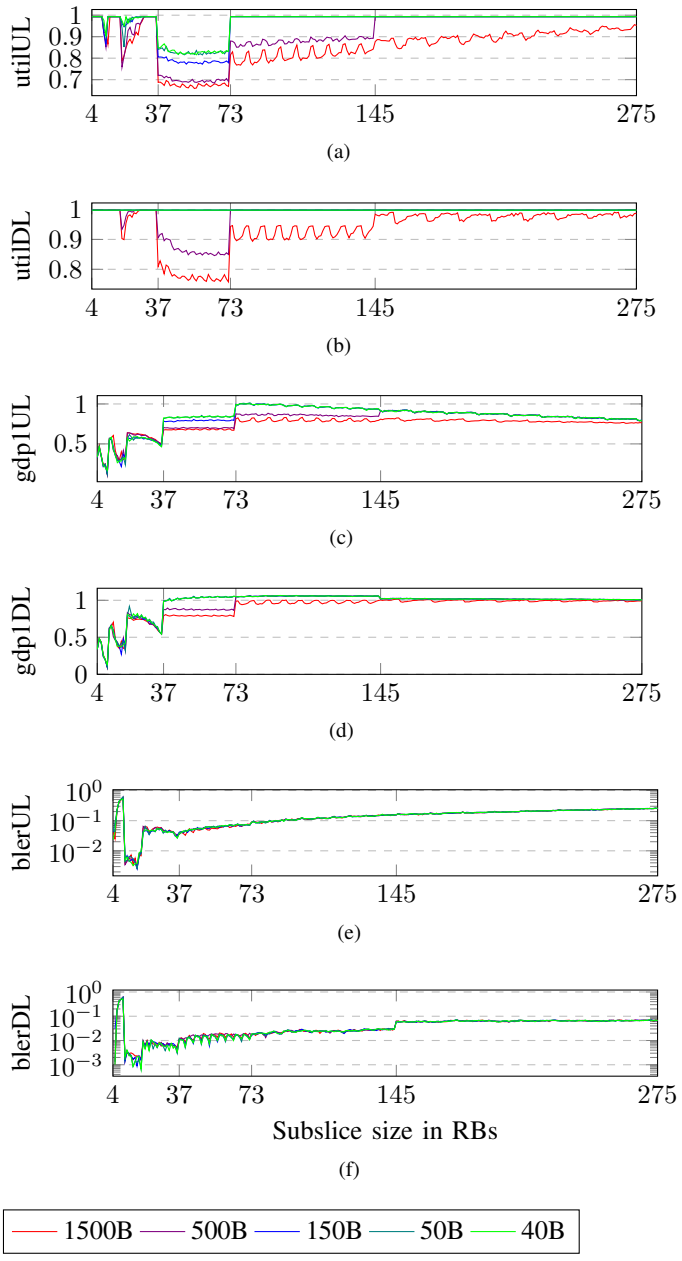

Fig. 1. Subslice performance data for all subslice sizes in range of 4-275 RBs, if packets of different sizes used. 6 KPIs: (a) bandwidth utilization in UL and (b)in DL, (c) goodput per one RB (Mbps) in UL and (d) in DL, (e) BLER in UL and (f) in DL.

B. Clusterability evaluation

Each data point in the dataset has values for six KPIs, subslice size in RBs, and packet size. To evaluate the clusterability [9] of subslice performance dataset, pairwise Euclidean distances were calculated for all data points. The histogram of all distances should be multimodal to expect the dataset to contain clusters. The histograms of pair-wise distances using combinations of KPIs are shown in Fig. 2. Histograms that characterize six or four KPIs look visually promising to

be three-modal; thus, these should be explored further. The subslice performance data could not be clustered into three clusters by seven variables, consisting of the subslice size and six KPIs. The subslice size variable is used for the recognition of a data point, similar to the packet size variable in the dataset.

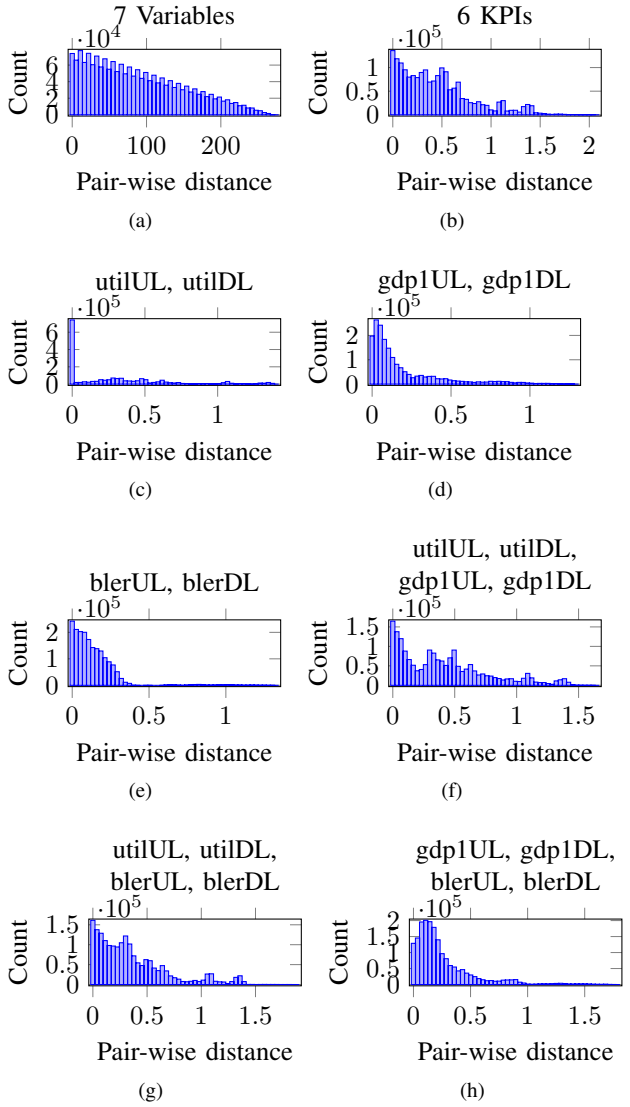

Fig. 2. The histogram of pair-wise distances of all data points in subslice performance dataset. Dimensions of each datapoint: (a) 6 KPIs and subslice size in RBs, (b) 6 KPIs, (c) bandwidth utilization in UL and DL, (d) goodput per one RB in UL and DL, (e) BLER in UL and DL, (f) utilization and goodput, (g) utilization and BLER, (h) goodput and BLER.

C. Evaluation of clustering results

Four types of clustering algorithms exist: density-based, distribution-based, centroid-based and hierarchical-based. One algorithm of each type was compared for suitability in clustering the subslice performance dataset into three clusters. K-means [10] is a centroid-based, simple, and popular clustering method. Gaussian Mixture Model (GMM) algorithm [11] is a distribution-based clustering method. Agglomerative hierarchy clustering algorithm [12] is a hierarchy-based clustering method. DBSCAN [13] is a density based method. The settings for DBSCAN are recommended in [14]: the minimum number of points, a parameter called $minPts$, and a small radius ϵ, that should contain the minimum number of points. Euclidean distance was used as the distance function. DBSCAN can identify outliers; however, a decision must be made for each data point.

To evaluate clustering results, three types of statistics were calculated. Davies-Bouldin index (DBI) [15] was calculated using

$$DBI = \frac{1}{k}\sum_{i=1}^{k}\max_{j\neq i}\{D_{i,j}\}, \tag{1}$$

where $D_{i,j}$ is the within-to-between cluster distance ratio for the ith and jth clusters. The smaller values indicate better clustering.

The Silhouette coefficient (SIL) [16] for each data point was calculated using

$$s_i = \frac{b_i - a_i}{max(a_i, b_i)}, \tag{2}$$

where a_i average distance of point i to the points in the same cluster, b_i is the minimum average distance of point i to the points of all other clusters. High positive value means highly separable clusters.

The third statistic calculated was sum of entropies (SEN) of clusters by using

$$E = -\sum_{i=1}^{k} p_i \cdot \log(p_i), \tag{3}$$

where k is a number of clusters, and p_i is the proportion of the points in the region i. Large values of entropy indicate poor clustering behaviour [17].

TABLE I
EVALUATION OF CLUSTERS. *DBI WAS CALCULATED AFTER OUTLIERS EXCLUDED.

Variables	DBI	SIL	SEN
k-means			
6 KPIs	0.779	0.715	0.833
utilUL, utilDL, gdp1UL, gdp1DL	0.694	0.742	0.833
utilUL, utilDL, blerUL, blerDL	0.608	0.753	0.768
gdp1UL, gdp1DL, blerUL, blerDL	0.483	0.848	0.504
GMM			
6 KPIs	0.944	0.608	0.649
utilUL, utilDL, gdp1UL, gdp1DL	1.20	0.696	0.571
utilUL, utilDL, blerUL, blerDL	0.536	0.723	0.398
gdp1UL, gdp1DL, blerUL, blerDL	0.461	0.861	0.457
Agglomerative			
6 KPIs	0.409	0.566	0.0850
utilUL, utilDL, gdp1UL, gdp1DL	1.60	0.490	0.471
utilUL, utilDL, blerUL, blerDL	0.771	-0.348	0.0850
gdp1UL, gdp1DL, blerUL, blerDL	0.248	0.783	0.0850
DBSCAN			
6 KPIs	0.675*	0.569	0.427
utilUL, utilDL, gdp1UL, gdp1DL	0.602*	0.753	0.586
utilUL, utilDL, blerUL, blerDL	0.415*	0.775	0.0662
gdp1UL, gdp1DL, blerUL, blerDL	NaN*	NaN	0.0767

Better clustering is indicated if DBI is small, SIL is high positive value and SEN is small. The results are presented in Table I for all 6 KPIs and different combinations of 4 of those KPIs. The DBI was the smallest when Agglomerative clustering was used. Reducing the number of KPIs improves DBI. The SIL was the highest when k-means was used. Reducing the number of KPIs increased the SIL, except in the case of Agglomerative. The SEN was the lowest when Agglomerative clustering was used. If the performance data were clustered by utilization and goodput KPIs or utilization and BLER KPIs, then DBSCAN had the best statistical values. If the performance data were clustered by goodput and BLER KPIs, then GMM and Agglomerative had the best statistics, but DBSCAN could not be evaluated.

Considering the statistics, the best clustering is possible with Agglomerative hierarchical clustering, followed by k-means clustering, DBSCAN, and GMM. If the number of KPIs is reduced, then the statistics are improved for k-means and GMM, especially when BLER is included in the KPIs.

D. Visualization of clustering results

The aim of the visualization is to confirm that too small, too large, and suitably sized subslices are in three separate clusters, and that some boundaries of subslice sizes between the clusters are visible. The clustering results are presented in Fig. 3. The clusters are numbered automatically, but matching of the cluster to the decision is performed manually. The cluster which contains the smallest subslice sizes is matched to "merge" cluster. The cluster which contains the largest subslice sizes is matched to "split" cluster. A cluster which contains other subslice sizes is "no change" cluster.

Both algorithms, k-means and GMM, have similar results (see Fig. 3a and 3b); however, k-means has subslices that are too small to be smaller than 37 RBs, whereas GMM has subslices that are too small with sizes of 7-9 RBs. The removal of BLER KPI did not significantly change the clustering result (see Fig. 3c and 3d); however, if goodput was removed, too small and too large subslices were in the same cluster (Fig. 3e and 3f. Agglomerative (Fig. 3i) and DBSCAN (Fig. 3j) had most of the data points in one cluster.

K-means had the second-best statistics, as shown in Table I, and it can identify subslices that are too small and suitable size better, as shown in the visualization of clusters in Fig. 3a and 3c.

IV. Learning the Clustering Results for a Classification

The k-means clustering results were selected for learning by a classifier. The classification results of the two input datasets are compared: all six KPIs and four KPIs (utilUL, utilDL, gdp1UL, gdp1DL). This set of four KPIs had values of statistics not worse than the set of 6 KPIs, and had meaningful results in visualization.

A. Classifier

The machine learning tools suitable for multi-class classification are k nearest neighbour (KNN), supporting vector

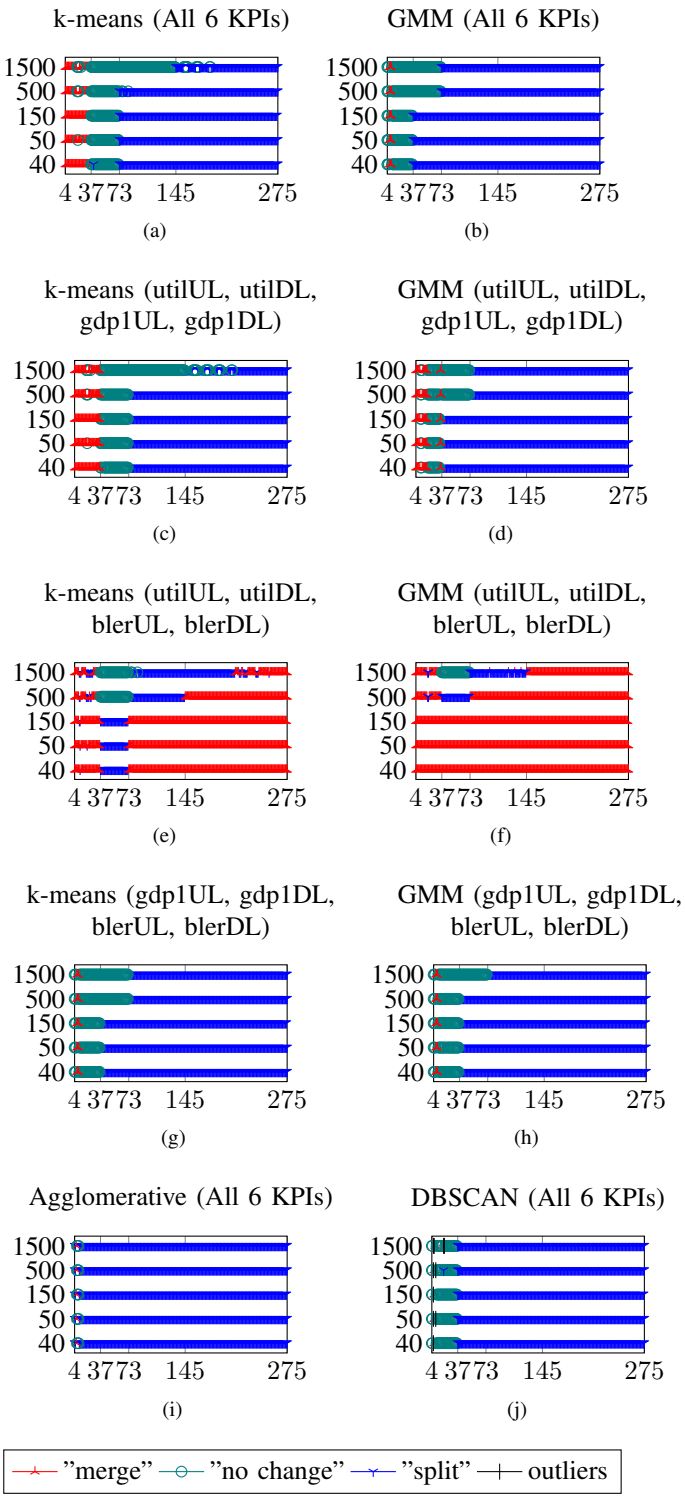

Fig. 3. Visualization of clustering results using different clustering algorithms and dimensions of a data point: (a) k-means (6 KPIs), (b) GMM (6 KPIs), (c) k-means (utilization and goodput), (d) GMM (utilization and goodput), (e) k-means (utilization and BLER), (f) GMM (utilization and BLER), (g) k-means (goodput and BLER), (h) GMM (goodput and BLER), (i)Agglomerative (6 KPIs), (j) DBSCAN (6 KPIs). The vertical axis shows used packet sizes in bytes, the horizontal axis shows subslice size in RBs.

machine (SVM) and neural network (NN). A neural network (NN) was selected as the tool for classification. NN has the flexibility and can work with large datasets.

The fully connected feedforward classifier NN consists of input layer, hidden layers, and output classification layer. The input layer contains 6 nodes, if 6 KPIs are used as inputs, and 4 nodes, if 4 KPIs are used as inputs. The output layer contains 3 neurons, one for each class ("merge", "no change", "split").

The rectified linear unit (ReLU) activation function is suitable for the neurons in the hidden layers of the classifier NN. The number of hidden layers and number of neurons in the hidden layer need to be found by trials. The number of neurons in the hidden layer, N_h can be determined using some hints from [18] and options are shown in Table II. One and two hidden layers were used in the trials, and the number of neurons in each layer was set in the ranges of 1-22. The training, validation, and testing set ratios were 0.8, 0.1, and 0.1, respectively.

TABLE II
OPTIONS FOR NUMBER OF NEURONS IN HIDDEN LAYER, N_h, IF KNOWN NUMBER OF INPUTS N_i AND NUMBER OF OUTPUTS $N_o = 3$

	Number of neurons in hidden layer	
Condition [18]	**6 KPIs**	**4 KPIs**
$N_o \leq N_h \leq N_i$	3, 4, 5, 6	3, 4
$N_h = \frac{2}{3} \cdot N_i + No$	5	5
$N_h < 2 \cdot N_i$	12	8

To evaluate the quality of classification, cross entropy losses were calculated for training, validation, and testing. Cross entropy loss shows the probability distribution difference between the predicted and true classes for the multi-class classifier. The cross entropy loss is calculated using

$$L = \sum_{j=1}^{n} \bar{w_j} \frac{\log m_j}{K \cdot n}, \tag{4}$$

where $\bar{w_j}$ are normalized weights, m_j is a classification score predicted for a true class, K is the number of classes, n is the number of data points.

The minimum number of neurons and layers with small losses of training, validation, and testing were selected.

B. Results of NNs

Two input datasets were considered. The k-means clustering results if 6 KPIs were used, and if utilization and goodput without BLER were used.

Each number of neurons in hidden layers are tested 100 times. Mean and confidence interval of 95% are calculated. The results of NN are shown in Fig. 4. The NN performance is evaluated by calculation of a cross entropy loss for training, validation and testing, respectively.

For input dataset which contained 6 KPIs, if the number of neurons is 7 or more, then the losses are not decreasing further. Both the training and validation losses are lower if one hidden layer is used than with 2 hidden layers. This means

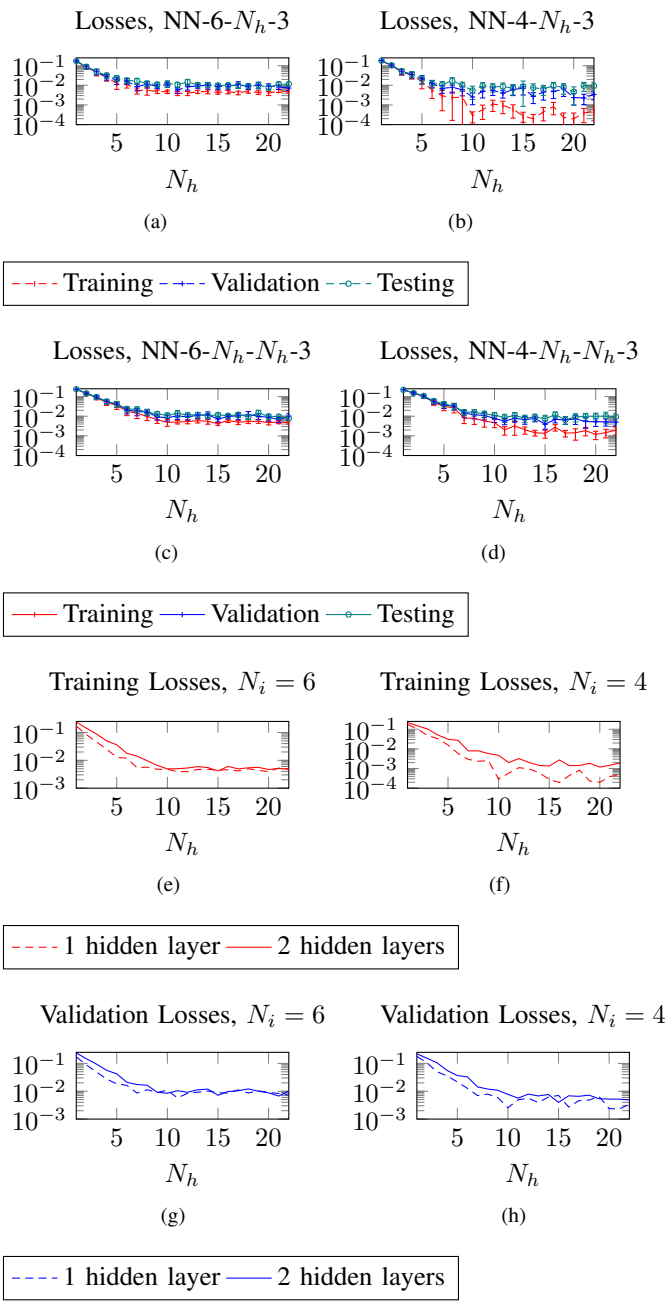

Fig. 4. Performance of a NN with one or two hidden layers (HL) and different number of neurons (N_h) in HL. (a) all losses for NN with one HL and input layer consisting of 6 nodes and (b) 4 nodes; (c) all losses for NN with two HLs and input layer consisting of 6 nodes and (d) 4 nodes, (e) training losses for NN with input layer consisting of 6 nodes and (f) 4 nodes; (g) validation losses for NN with input layer consisting of 6 nodes and (h) 4 nodes.

that one hidden layer with 7 neurons is sufficient configuration of classifier NN for subslice performance data with subslice sizes in the range of 4-275 RBs. The NN settings for a dataset of six KPIs are NN-6-7-3. The notation used, is the number of neurons in the input layer (6), the number of neurons in each hidden layer (1 hidden layer with 7 neurons), and the last number is the number of neurons in the output layer (3).

For input dataset which contained 4 KPIs, if the number of neurons is 10 or more, then the losses are not decreasing further. Similarly, all losses are lower when one hidden layer is used. The NN settings for a dataset of 4 KPIs are NN-4-10-3.

V. Experiments

The constructed classifier, NN, was evaluated using a dataset of smaller subslices. Given the data of the six and four KPIs, the constructed NN with one hidden layer consisting of 7 and 10 neurons, respectively, determined the action for the subslice. The results are shown in Fig. 5. Generally, smaller subslices are classified as "merge" and a few larger subslices are "split". If fewer KPIs are given as inputs, then some subslices smaller than 37 RBs are classified as "no change". When using 1500-byte packet size then the subslice size range of "no change" is larger than if shorter packets are used.

It should be noted that these subslices were created by clustering UEs using k-means by UE BLER, and RBs were allocated proportionally to the number of UEs in the cluster. Second, the classifier NN does not know the subslice size in the RBs. Thus, the results are acceptable, and better classification was achieved if the input dataset contained six KPIs.

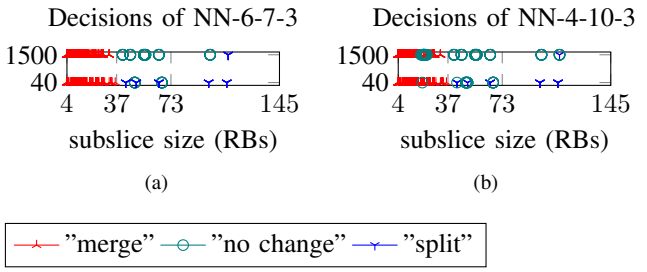

Fig. 5. Classification experiments for smaller subslices using: (a) NN-6-7-3, (b) NN-4-10-3. The vertical axis shows used packet sizes in bytes, the horizontal axis shows subslice size in RBs.

VI. Conclusion

We have investigated the decision to modify slices using ML tools. The slice performance data is clustered into three clusters to match the slice or subslice configuration decisions. The clustering results were used as training data for the neural network, which determines the slice modification action based on the slice performance. The KPIs used as inputs were bandwidth utilization, goodput per allocated RB, and slice BLER, all for UL and DL. None of these KPIs can be omitted. This mechanism can help automatically decide subslices configuration. Further work is required to investigate the subslice performance dependence on the subslice size if the UEs are not in a good signal coverage.

Acknowledgment

This work has received funding partly from the European Union's Horizon 2020 Research and Innovation Program under Grant 951867 '5G-ROUTES' and Grant 101058505 '5G-TIMBER'. This work in the project "ICT programme" was supported by the European Union through European Social Fund, and TAR16013 Center of Excellence 'EXCITE IT'.

References

[1] GSM Association, "Generic Network Slice Template Version 8.0," Tech. Rep., 2023, pp. 1–72, [Online]. Available: https://www.gsma.com/newsroom/resources/ng-116-generic-network-slice-template-v8-0/.

[2] 3GPP, "TS 38.211 - NR; Physical channels and modulation," 2020, [Online]. Available: https://www.3gpp.org/DynaReport/38211.htm.

[3] 3GPP, "TS 28.535 - Management and orchestration; Management services for communication service assurance; Requirements," Tech. Rep., 2020, [Online]. Available: https://www.3gpp.org/DynaReport/28535.htm.

[4] S. Ravindran, S. Chaudhuri, J. Bapat, and D. Das, "Isolation-based Sub-Slices for Throughput Optimization in 5G Radio Access Network," in *2019 IEEE Int. Conf. Electron. Comput. Commun. Technol.*, IEEE, Jul. 2019, pp. 1–6, DOI: 10.1109/CONECCT47791.2019.9012928.

[5] S. K. Singh, M. M. Salim, J. Cha, Y. Pan, and J. H. Park, "Machine Learning-Based Network Sub-Slicing Framework in a Sustainable 5G Environment," *Sustainability*, vol. 12, no. 15, p. 6250, Aug. 2020, DOI: 10.3390/su12156250.

[6] K. Hooli, P. Kinnunen, E. Tiirola, *et al.*, "Extending 5G to narrow spectrum allocations," *IEEE J. Sel. Areas Commun.*, pp. 1–1, 2023, DOI: 10.1109/JSAC.2023.3273703.

[7] MathWorks, *NR Cell Performance Evaluation with Physical Layer Integration*, 2022, [Online]. Available: https://se.mathworks.com/help/5g/ug/nr-cell-performance-evaluation-with-physical-layer-integration.html (visited on 01/19/2022).

[8] M. Kulmar, I. Müürsepp, and M. M. Alam, "Heuristic Radio Access Network Subslicing with User Clustering and Bandwidth Subpartitioning," *Sensors*, vol. 23, no. 10, p. 4613, May 2023, DOI: 10.3390/s23104613.

[9] A. Adolfsson, M. Ackerman, and N. C. Brownstein, "To cluster, or not to cluster: An analysis of clusterability methods," *Pattern Recognit.*, vol. 88, pp. 13–26, Apr. 2019, DOI: 10.1016/j.patcog.2018.10.026.

[10] J. MacQueen, "Some Methods for Classification and Analysis of MultiVariate Observations," in *Proc Berkeley Symp. Math. Stat. Probab.*, 1965, pp. 281–297.

[11] D. Reynolds, "Gaussian Mixture Models," in *Encycl. Biometrics*, Boston, MA: Springer US, 2015, pp. 827–832, DOI: 10.1007/978-1-4899-7488-4_196.

[12] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, Wiley Seri. Wiley, 2009, ISBN: 9780470317488.

[13] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *Proc. 2nd Int. Conf. Knowl. Discov. Data Min.*, 1996.

[14] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN Revisited, Revisited," *ACM Trans. Database Syst.*, vol. 42, no. 3, pp. 1–21, Sep. 2017, DOI: 10.1145/3068335.

[15] D. L. Davies and D. W. Bouldin, "A Cluster Separation Measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979, DOI: 10.1109/TPAMI.1979.4766909.

[16] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, Nov. 1987, DOI: 10.1016/0377-0427(87)90125-7.

[17] C. E. Shannon, "A Mathematical Theory of Communication," *Bell Syst. Tech. J.*, vol. 27, no. 4, pp. 623–656, 1948, DOI: 10.1002/j.1538-7305.1948.tb00917.x.

[18] J. Heaton, *Introduction to Neural Networks with Java*. Heaton Research, 2008, ISBN: 9781604390087.