Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Performance evaluation of Fuzzy-logic-based algorithms for inter-slice handover management in 3GPP architectures

Döme Matusovits

mdome@hit.bme.hu

Budapest University of Technology and Economics

László Bokor

Budapest University of Technology and Economics

Research Article

Keywords: Fuzzy logic, inter-slice handover, algorithms, handover management, performance evaluation, 3GPP 5G/6G architectures

Posted Date: November 26th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10830640/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Performance evaluation of Fuzzy-logic-based algorithms for inter-slice handover management in 3GPP architectures

Döme Matusovits (corresponding author)[1] and László Bokor[1,2]

[1] Department of Networked Systems and Services, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111 Budapest, Hungary
[2] HUN-REN-BME Cloud Applications Research Group, Magyar Tudósok Körútja H-1117 Budapest, Hungary
mdome@hit.bme.hu, bokorl@hit.bme.hu

Abstract. Network slicing is a new technology in 5G/6G networks that partitions the mobile network into distinct segments to continuously meet specific service requirements. This scheme also introduces a new handover type, called inter-slice handover, in which the user switches between resources occupied by different slices, rather than just between base stations. Essentially, users or user flows will be handed over to another slice, meaning switching between shared and dedicated radio and core network components. The handover between the resource-realized slices is based on the mobile user's actual need and the current network state. Due to the above, inter-slice handover is inherently complex, with multiple trigger types and possible handover scenarios, and can occur even in the absence of any physical UE mobility. For that reason, appropriate schemes should be designed to handle the increased complexity of such special handover situations. In this paper, we propose two algorithms for inter-slice handover management, based on Mamdani-type Fuzzy logic and Adaptive Neuro-Fuzzy Inference System (ANFIS). We examined our proposed solutions for inter-slice handover cases, focusing on the degradation of the monitored slice's KPIs.

Keywords: Fuzzy logic, inter-slice handover, algorithms, handover management, performance evaluation, 3GPP 5G/6G architectures

1 Introduction

The advanced use cases – such as autonomous driving, remote surgery, Virtual/Augmented Reality (AR/VR), and manufacturing automation – that modern telco infrastructures are expected to support require strict Service Level Agreements (SLAs) for communication. With the advent of next-generation mobile networks, efficient SLA provisioning can be met through an innovative architectural approach called Network Slicing (NS). This approach divides the entire mobile network into individual segments to ensure service guarantees. During the operation of these segments (i.e., network slices), there may be cases when a user needs to change the slice they are currently using. This is when inter-slice handover takes place. Unlike traditional handovers,

which involve physical mobility between base stations, inter-slice handovers occur between virtual network resources. Switching between slice resources introduces additional scenarios during the handover procedure. Inter-slice handover can occur even without UE mobility; it depends on whether the radio station is shared or dedicated between the two slices during handover. Of course, other segments of the slices are also crucial in this matter, but from an inter-slice handover perspective, the base stations remain the endpoints of the network. The whole slicing approach introduces additional trigger types, so there is no question that inter-slice handovers are complex. We have already made a brief overview of handover decision and selection algorithms in our previous paper [1]. Based on that, we decided to examine AI-based approaches, particularly fuzzy and ML-based solutions, in depth. In this paper, we will introduce these algorithms and their contribution to our inter-slice handover management framework [1][2]. We examine how these complex approaches perform in such complex mobile network use cases. Also, we compare the proposed algorithms in terms of decision efficiency. Moreover, we provide an overview of how these inter-slice handover decision frameworks could fit into a 3GPP-compliant test environment. That means we give explicit associations between the testbed and the algorithm components to the appropriate 3GPP piece elements. It is an important aspect from later standardization perspective.

The paper is organized as follows. We summarize the concept of network slicing, including the evolution of technologies and standards, as well as future directions for NS, in Section 2, along with the theoretical background of the slice handover algorithms examined. After that, we describe the proposed framework, our testbed, the implementation details, the introduced inter-slice handover workflow, the test scenarios applied, and the test results for the evaluated fuzzy-based algorithms in Sections 3 and 4, respectively. Last, we conclude the paper and propose future work in Section 5.

2 Evolution of Network Slicing and inter-slice handover management within 3GPP and ETSI standardization

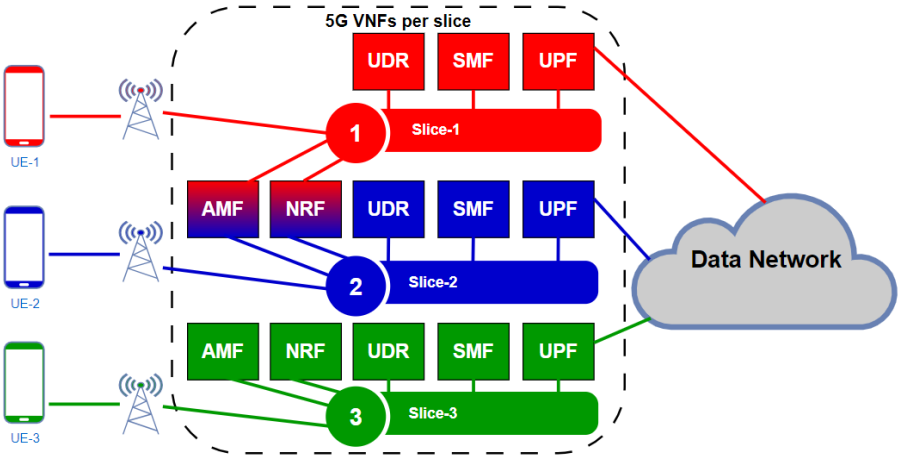

Fig. 1. The network slicing paradigm

Network slicing is a term proposed by 3GPP to enable individual, virtualized segments within next-generation mobile networks. It was first defined in [3]. Resources that can be shared or dedicated across slices include Network Functions (NFs) from the core side and base stations from the radio side (**Fig. 1**). For example, the Access and Mobility Management Function (AMF) is shared between slices 1 and 2. However, the User Plane Functions (UPFs) are dedicated to both slices. At the lower level, all resources are unique to each slice. These granular components are the network

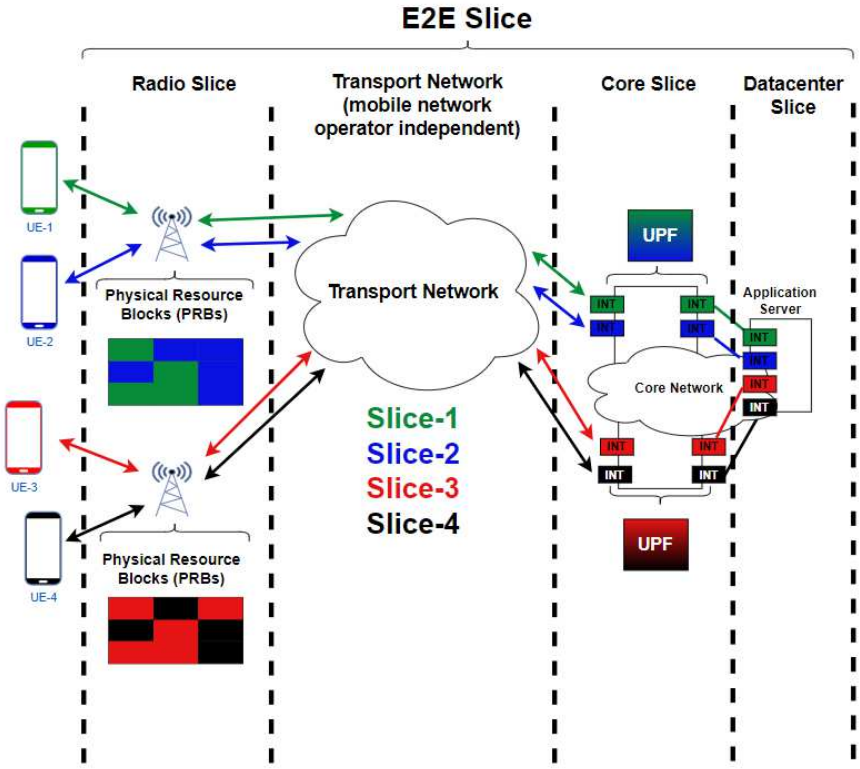

Fig. 1. Low-level realization of network slicing

interfaces (INTs) and the Physical Resource Blocks (PRBs). That means each high-level component has unique subresources, regardless of whether they are shared or dedicated to a particular slice **(Fig. 2)**. This is an essential aspect of network slicing because, to meet service requirements, dedicated network resources must ultimately be allocated. Network slicing encompasses all network segments, including the radio, transport, core, and even data center networks. However, different standardization organizations are responsible for different segments:

- Radio (ITU-R, IEEE, 3GPP)
- Transport (IETF)

- Core (3GPP, ETSI)
- Datacenter (IETF)

In this paper, we will focus primarily on the radio and core segments and specifically concentrate on the 3GPP and ETSI standards. It's not an obvious task to ensure synergistic cooperation between network segments. Mainly because mobile operators don't influence the transport and data network, where the application is hosted. As a consequence, if any networking-related issue had arisen at the transport or data network level, it couldn't be handled independently of ensuring that the other segments are successfully operational. For this reason, integrating network slicing into 5th-generation mobile networks is a challenging engineering task. Of course, there are partial workarounds to cover the data center segment at least by migrating the application server to the core network, as shown in **Fig. 2**. This also entails a new service model that takes responsibility for the application and its connectivity under a desired Service Level Agreement (SLA).

Network slices are also orchestrated just like the Virtualized Network Functions (VNFs), networking, and computing infrastructure of 5G mobile networks. This orchestration framework is defined by 3GPP [4][5] and ETSI [6]. The key components are all related to the whole lifecycle management of network slices (**Fig. 3**):

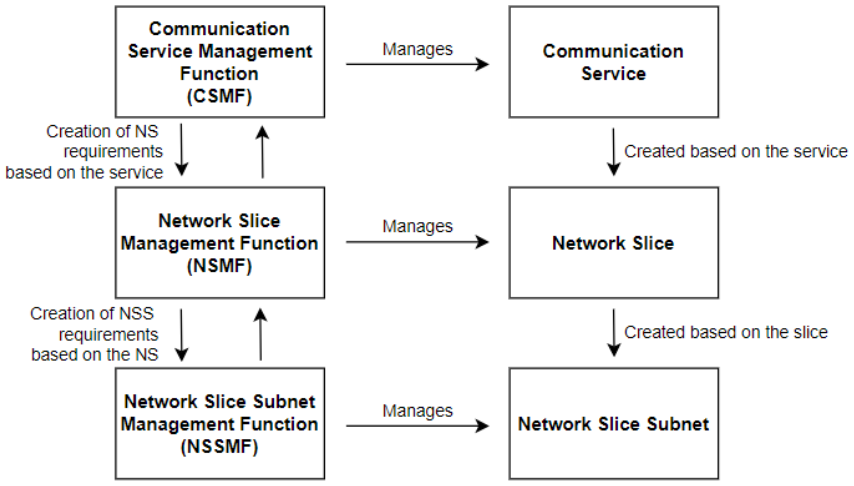

Fig. 2. Orchestration of network slices

- *Communication Service Management Function (CSMF):* is the highest level of service instantiation, mainly for service and lifecycle management.
- *Network Slice Management Function (NSMF):* responsible for the orchestration and lifecycle management of network slice instances, which are part of the Communication Service.

- *Network Slice Subnet Management Function (NSSMF):* the lowest level of communication service hierarchy where a particular slice subnet is managed as part of a network slice.

There may be cases where a mobile user wants to change their attachment from one slice to another. This is called inter-slice handover. Unlike traditional physical mobility scenarios, this handover type occurs virtually between slices. Depending on the base station allocation for the slices, the handover can happen within a base station or between two distinct radio instances. The possible triggers for handing over to a new slice are the following:

- Slice is not available anymore
- Slice is under maintenance
- Slice subscription expiry
- User preferences
- Security considerations
- Degradation of the KPI metrics
- etc.

We use different algorithms for handling these trigger types. In our previous paper [1], we already categorized these approaches. However, those are collected for vertical handovers, and our use case will be restricted to inter-slice handover scenarios. Also, in paper [1], the focus was on collecting algorithms for making handover decisions only, ignoring the network selection entirely. To evaluate a handover decision algorithm's efficiency, we also need to check whether the selected network yields better results according to the examined criteria. That's why we also included scenarios where the decision and network selection aren't decoupled at all. Based on this consideration, we summarized a list of the most remarkable algorithms ([7][8][9]) with a possible pairing of the trigger types:

- *Cost-based decisions (User preferences, Degradation of KPI metrics):* These algorithms are based on the costs of potential target networks and account for user preferences. Different weights are assigned to examined attributes. Then, these input metrics are fed into a cost function, which calculates a result and makes the handover decision and network selection accordingly.
- *Context-based (User preferences, Security considerations, Slice is not available anymore, Slice is under maintenance):* The handover decision is made by gathering information from neighboring nodes/networks and selecting a new network accordingly.
- *Multiple Attribute Decision Making (MADM) (User preferences, Degradation of KPI metrics, Security considerations):* It aims to select a candidate network from a set of available networks based on multiple criteria. The most popular MADM techniques are Simple Additive Weighting (SAW), Analytic Hierarchy Process (AHP), Multiplicative Exponent Weighting (MEW), Technique

for Order of Preference by Similarity to Ideal Solution (TOPSIS), and Gray Relational Analysis (GRA).

- *Markov-based decision algorithms (Degradation of the KPI metrics, Slice subscription expiry, Slice is no longer available):* Markov decision schemes are dynamic processes in which the decision epochs follow a probability distribution. That means outcomes are determined by probability and the decision maker's choices.
- *Intelligence-based methods (Degradation of KPI metrics):* Intelligence-based algorithms are mostly AI-based, including Machine Learning (ML) and Fuzzy-based schemes. We can solve complex decision-making problems with such solutions.

Our proposed algorithms can both make handover decisions and select new slices in response to changes in KPI metrics. KPI degradation is a complex problem in itself, and an appropriate algorithm should be proposed to detect changing metric values and reflect them in the current slice's state. For this reason, we have begun to examine fuzzy and ML-based inter-slice handover management frameworks.

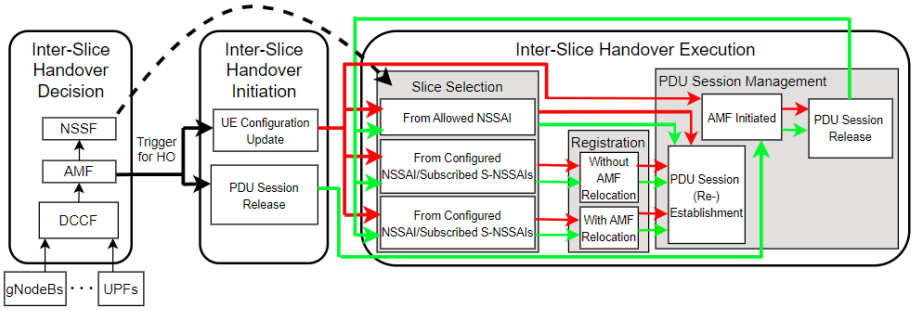

Fig. 3. 3GPP-based Inter-slice handover mechanism

To our knowledge, there is no such ETSI or 3GPP-based standardization for inter-slice handover. However, by looking at the components piece by piece, we can infer the whole picture. We have already provided a brief overview of the 3GPP-related inter-slice handover procedures in paper [1]. Based on that, we have developed a more general, recapitulated approach to the 3GPP-related procedures depicted in **Fig. 4**. Of course, mobile network standardization barely addresses algorithms. However, we can conclude their possible placement within the 3GPP architecture based on the standardized mechanisms we have studied so far. This topic will be addressed in chapters 3 and 3.2.

6G standardization is still in its early stages, also meaning that standardization for 6G network slicing has not yet formally begun in the 3GPP releases [10]. However, it will likely evolve directly from the 5G slicing architecture rather than entirely replacing it. There are already several research papers discussing potential new application areas, and network slicing is expected to evolve alongside the emergence of these new use cases:

- *AI-Native Network Slice Management:* Future research focuses on embedding AI/ML-based solutions directly into network slice lifecycle management. That includes AI-assisted slice creation [11][12], predictive resource allocation [11][12][13][14][15], zero-touch service management [14], and self-optimization slices [12][14][16]. Of course, it's worth considering AI-based solutions for inter-slice handover decision-making and new network slice selection, within our research scope for 5th-generation mobile networks. Also, one existing paper mentioned this issue and proposed AI-based algorithms for 6G inter-slice handover prediction ([17]).
- *Cross-Domain Slice Orchestration:* This is essentially the same issue as outlined in Section 2 for end-to-end (E2E) network slicing. In current architectures, slicing is interpreted separately in the radio, transport, and core networks. Furthermore, as previously mentioned, the concept of slicing cannot be strictly applied to the transport network, as this domain typically falls outside the mobile operator's purview. Nevertheless, some research papers on 6G network slicing [17][18][19] extend slicing orchestration to the transport network at a high level. While this would indeed represent the ideal form of E2E slicing, existing studies generally do not explain how such an approach could be implemented in practice, nor why sixth-generation mobile networks would be able to resolve this issue. However, a paper already exists that outlines the problem ([14]). Also, some examples propose a more realistic slicing framework, in which only the radio and core slices exist ([20][21]).
- *Digital twins for NS:* By creating counter-twins of physical network domains, real-time simulation/emulation of network slices can be achieved. This can be used to realize the network behaviour in the virtual world. Based on that, we can make precise KPI predictions, which are essential for slice provisioning, as discussed in papers [15] and [18].
- *Security of NS:* Although it is not a feature that will be automatically introduced in the 6G era, it is a required aspect for implementing 6G architectures. The 5G network slicing security architecture follows the zero-trust paradigm, requiring continuous verification of all entities accessing a network slice and granting only minimal privileges by default. However, inter-slice handover poses security breaches as discussed in paper [23]. Also, given the 6G vision of expanding intelligent virtualization to the edge and the use of multi-tenant clouds in 6G RAN, future slicing frameworks may become more vulnerable to cyberattacks due to heterogeneous technologies, insecure practices, and limited economic incentives for robust security solutions [14][21]. Therefore, several research challenges must be addressed to enable end-to-end zero-trust security in 6G network slicing.
- *Energy-saving considerations in NS:* 6G mobile networks are expected to consume more energy than previous generations. On the radio side, the base stations will operate in higher-frequency bands (THz), leading to higher power consumption. Introducing energy-saving methods is crucial to sustainability and operational cost reduction. This leads to a tradeoff between QoS guaran-

tees and energy saving for the RAN slicing scheduler to decide. There is ongoing research on this topic, such as highlighted in paper [22], which proposes a meta-reinforcement learning (MRL) method to optimize the long-term average energy consumption of the RAN.

3 The proposed framework

3.1 A 3GPP-compatible inter-slice handover scheme based on Mamdani Fuzzy Systems

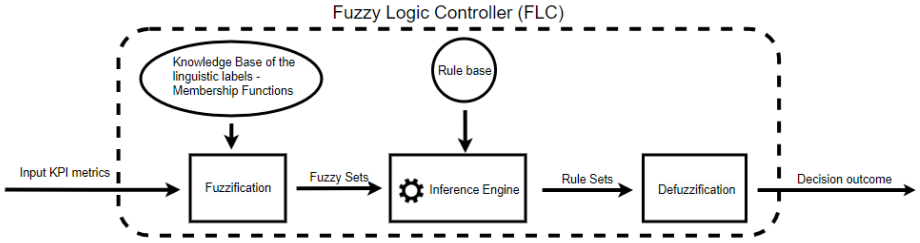

Fig. 4. Architectural overview of Fuzzy Logic Controllers (FLC)

We have already discussed the essential working of fuzzy systems in paper [1][2]. In this section, we summarize these concepts. Fuzzy logic is a mathematical approach for handling imprecision and nonstatistical uncertainty. It enables the processing of vague and uncertain information, converting it into a form that supports human reasoning. In contrast to Boolean logic, it transitions on a scale from entirely false to completely true. This is a key point in Fuzzy Logic Controllers (FLC), as it enables the smooth scaling of decision certainty. There are different fuzzy logic approaches. A general, well-known one is the Mamdani-based fuzzy logic systems. It has three main stages, which are evaluated in sequence (see **Fig. 5)**:

- *Fuzzification:* The input parameters are fed into a so-called fuzzifier. This means the real-time measured KPIs are transformed into fuzzy sets. Each metric has its own possible range of fuzzy set selection to which it can be grouped. These are elements with varying degrees of membership in a set. They are represented as values that are obtained from a membership function. The membership function takes the aforementioned metrics as input and produces the corresponding fuzzy sets (see the example in **Fig. 6**). The membership functions overlap, which is essential when discussing partial truth. This allows a metric to be assigned to more than one fuzzy set. Each subset has a degree of membership ranging from 0 to 1. Different kinds of membership functions exist; these are the most common ones [24] (the names refer to the function's shape):
    - Triangular

- Trapezoidal
- Gaussian
- Bell-shaped

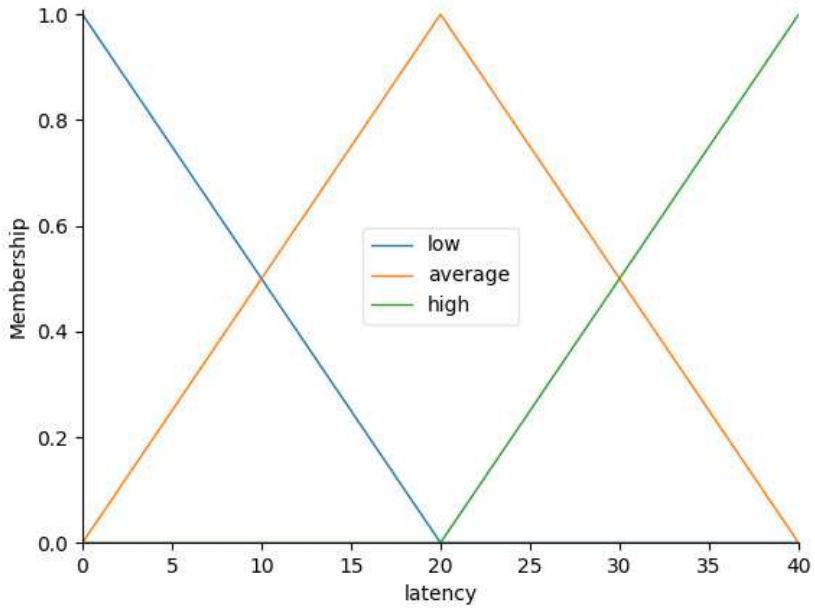

Fig. 5. Overview of membership functions

- *Inference Engine:* In this stage, the inference engine evaluates fuzzy rules defined in the rule base. The rule entries specify a specific pairing between certain fuzzified outputs and the given input sets. The input fuzzy sets configured in the entries are referred to as *antecedents*. The evaluated output sets based on the rules are the *consequents*. Usually, these rule entries are created manually. It is advantageous because we don't have to train the algorithm, unlike with other ML models. On the other hand, we must always adopt the rule-based approach if network conditions change rapidly. This is a limitation of Mamdani-based fuzzy systems.
- *Defuzzification:* The fuzzified outputs are transformed back into crisp values, enabling the controller to make an exact decision. The calculated value lies within the configured consequent interval. Depending on how the fuzzified outcomes are processed, different defuzzification methods exist [24]. In paper [2], we can see an example of the defuzzification workflow. The most common defuzzification methodologies are the following:
    - *Centroid:* The most commonly used approach for defuzzification (in this work, we also used this technique). It returns the center of gravity, which means that the area under the fuzzy sets is weighted by

their center positions (on the x-axis). These areas are summed and then divided by the original sum of the areas (without weighting). Based on the calculation, we will obtain a crisp output value on the x-axis.

- *Weighted Average:* It differs from the centroid method in that we weight each fuzzy set's membership value by the mean x-value rather than by the area under the membership function.
- *Max Membership Method:* The highest value of the membership function will be the produced crisp value. Commonly used in the case of triangular functions, where the maximum value is located at a peak point.
- *Mean-Max Membership:* With respect to the x coordinates position, the mean x value is determined based on where the output membership function reaches its maximum. In the case of a trapezoidal function, there is a plateau with minimum and maximum x coordinates, where the maximum membership value exists. In contrast, a triangular function has one peak maximum value, so that the mean value will be the same, just like with the Max Membership Method.

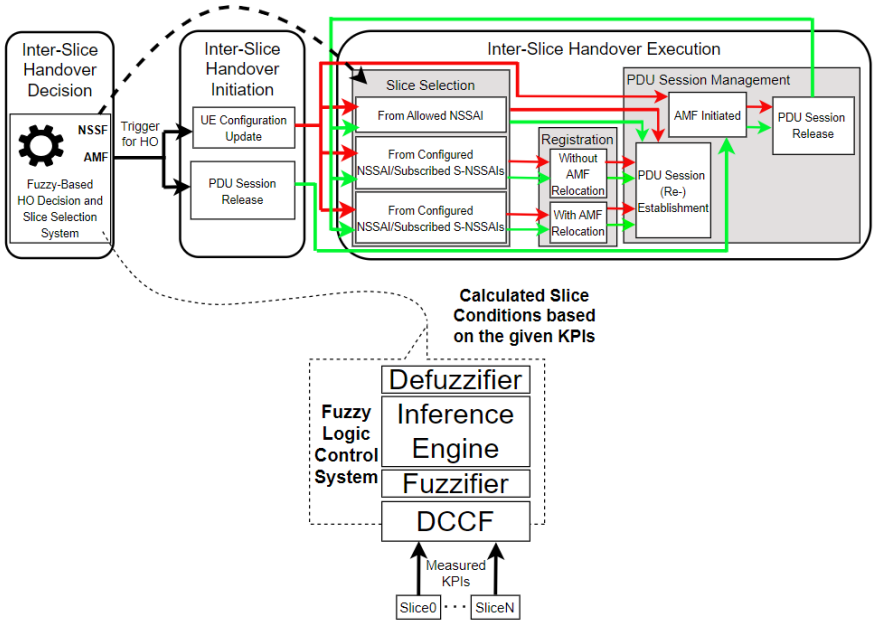

Fig. 6. Our proposed framework to realize Mamdani-based fuzzy algorithms into the 3GPP-based inter-slice handover scheme

As far as we are aware, the following research papers [25][26][27][28][29] use Mamdani-based fuzzy algorithms for inter-slice handover management. For exclusive decision-making, only paper [27] addresses both inter-slice handover decision and new

slice selection; the existing papers [25][26][28][29] do not. Our fuzzy logic-based frameworks for both Mamdani and ANFIS algorithms can make inter-slice handover decisions and select new slices as well. During evaluation, both decision and selection mechanisms are important for a full picture of handover efficiency. Moreover, we emphasize that none of the above-mentioned research studies considered 3GPP-compliant procedures in their architectural overviews of inter-slice handovers, in contrast to our work. Besides the general overview of 3GPP-compliant working of inter-slice handovers, we specifically consider a 3GPP overview of realizing Mamdani-based inter-slice handover decision frameworks depicted in **Fig. 7**. The key point here is that the AMF is the NF making inter-slice handover decisions based on the monitored KPI metrics collected by the Data Collection Coordination Function (DCCF). Besides, the Network Slice Selection Function (NSSF) will decide which slice to select.

3.2 A 3GPP-compatible inter-slice handover scheme based on Adaptive Neuro-Fuzzy Inference System (ANFIS)

Basically, ANFIS [24] [30] [31] is a special type of fuzzy system in the sense that its three main decision stages (fuzzification, rule evaluation, defuzzification) are manifested in a Neural Network (NN). In general, it consists of five layers, each with its own functional segment, depicted in **Fig. 8**.

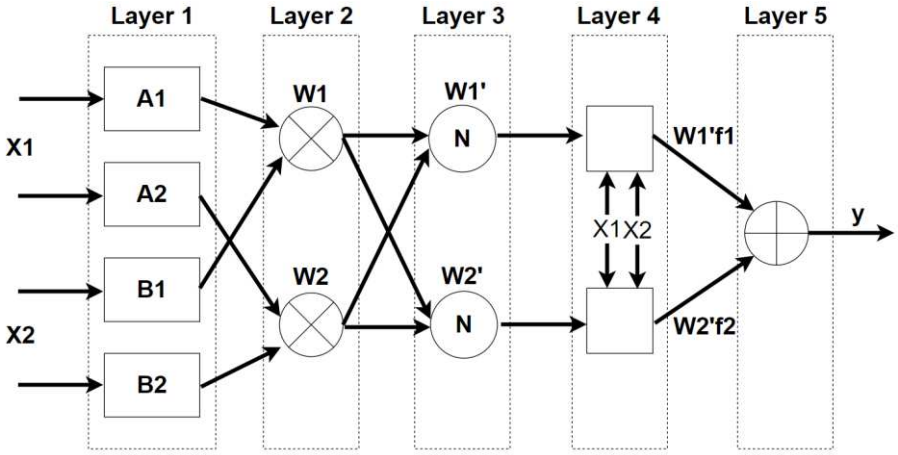

Fig. 7. Structure of ANFIS

In the first layer, the nodes represent membership scores computed from the input metrics (x1, x2). Essentially, this layer represents the fuzzifier. We can see that two distinct membership functions are configured, which manifest the proper fuzzy sets and produce the scores (A1, A2, and B1, B2, respectively). We refer to these produced values as premise parameters, as they will be optimized during the learning process. The rule evaluation is implemented by Layers 2, 3, and 4, respectively. The second Layer calculates the accumulated firing strength of rule antecedents. This is achieved through a so-called t-norm operation [32]. It is a binary operation specifically used in fuzzy logic-

based frameworks. It computes the intersection of the given fuzzy sets based on specific algebraic properties. For example, the 'and' operation takes the minimum of the membership scores, while the 'or' operation takes the maximum membership score. In the third Layer, we normalize the resulting weights (w1, w2) and use them to compute each rule's consequents (w1', w2'). This is done by the fourth Layer, where we use defined functions with respect to the normalized weights and the consequent parameters (pi, qi, ri) (1):

$$O4=wi'fi=wi'(pix1+qix2+ri) \quad (1)$$

There can be different kinds of functions applied to this task. Usually, linear functions are used for their simplicity (the equation above (1) is linear). Of course, other alternatives can also be considered, especially in cases where the input-output relation is remarkable. In these situations, the use of activation functions [33] is recommended. In the last layer, we calculate the linear combination of the consequent outputs represented by a single node (2):

$$O5 = \sum_i wi'fi \quad (2)$$

At this point, we must emphasize that the crisp output is generated directly from the rule consequents (via the functions used in Layer 4). That means defuzzification doesn't really exist in ANFIS systems. The idea behind structuring the inference system with this approach is the *Takagi-Sugeno-Kang Fuzzy Inference System (TSK FIS)* [34].

The given data is labeled in each case, making this a typical supervised ML regression task, where *we estimate the output using* a selected function. Two types of learning algorithms are used in ANFIS [31]. These methods are created by Jang, the inventor of ANFIS [35]:

- *Hybrid (Offline):* It optimizes the antecedents by executing two basic workflows during the process: a forward pass and a backward pass. In the forward pass, the functional signal from Layer 1 is passed directly through the ANFIS network to Layer 4, where the consequent parameters (function's variables) are calculated by the least squares estimate (LSE). At this point, the premise parameters from Layer 1 remain fixed. The backward pass procedure starts after computing the error. It is usually given by the Root Mean Square Error (RMSE) loss for the output. During this process, the consequent parameters are kept unchanged while the premise parameters are updated using the Gradient Descent method. In optimal cases, the error converges as we iterate through each epoch. That's how the input membership functions are optimized; this is what we should do by hand when we set up a Mamdani-based fuzzy system. In general, this learning method is the recommended approach, as it converges faster and is well-suited to linear functions. We also employed this approach when training the ANFIS.
- *Online:* During online learning, all parameters are updated and optimized using the Gradient Descent method. Online learning differs from the hybrid approach because, while the hybrid method trains the entire dataset per epoch (an iteration of the entire dataset to teach a model), online training updates

sample-by-sample. During online learning, a so-called instantaneous error is computed in the backward pass, meaning it is calculated for each data sample as it is processed.

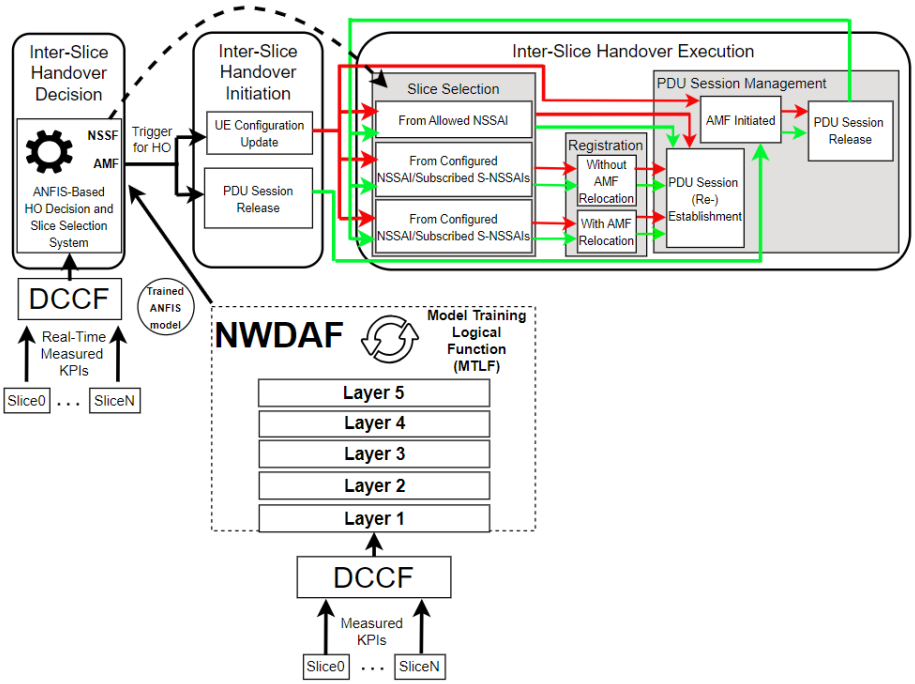

Fig. 8. Our proposed framework to realize the ANFIS algorithm in the 3GPP-based inter-slice handover scheme

In the 3GPP architecture, ML models are trained via the NWDAF [36], where the DCCF provides 5G traffic data. This is essential when using ANFIS, since it relies on an ML model as its core decision engine. The trained model can be used by the AMF to make inter-slice handover decisions, as illustrated in **Fig. 9**. This schematic illustrates our ANFIS-based handover decision framework, incorporating 3GPP-compliant components and procedures. The only difference from the Mamdani-based 3GPP-compliant interworking (depicted in **Fig. 7**) is the additional placement of the NWDAF NF. To the best of our knowledge, no such paper has proposed an ANFIS-based algorithm for inter-slice handover management.

4 Performance Evaluation

4.1 Testbed

The testbed was based on our initial research papers [1][2]. Since then, we have extended and revised some aspects of the testbed architecture. Initially, our proposal addressed only inter-slice handover decision functionality. In this paper, we additionally included an optional configuration to select the network slice using a fuzzy-based algorithm. In this way, we can examine the inter-slice handover decision efficiency in a

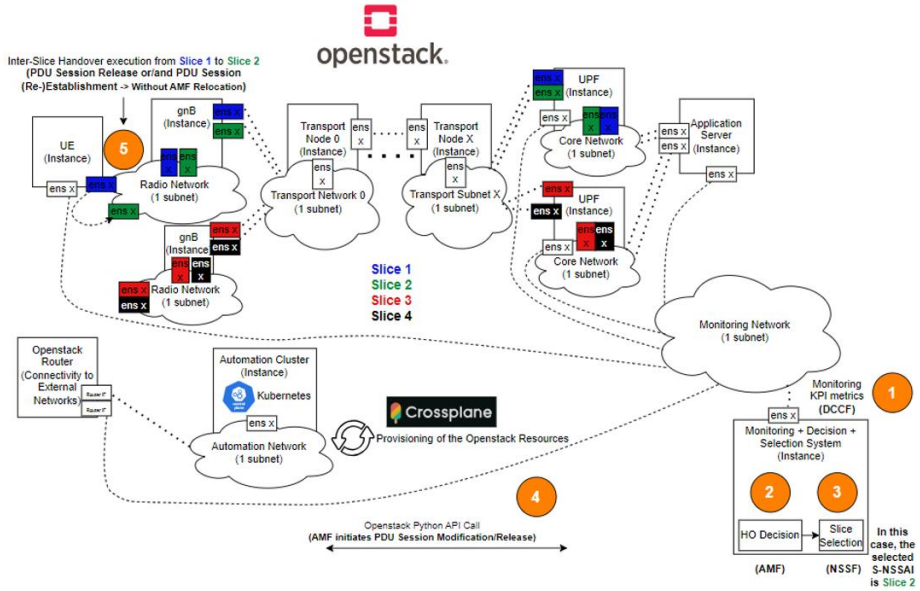

Fig. 10. Low-level testbed design

, wider scope. Additionally, the low-level design of our testbed was modified to better align with our proposed architecture (**Fig. 10**).

We instantiated four network slices for the measurement: Slice1, Slice2, Slice3, and Slice4. For each slice, we placed one UE (UE0, UE1, UE2, or UE3) to generate traffic on that slice. The referenced user is the UE0, which was examined during the measurement. The whole network setup consists of 4 UEs, 2 gNodeBs, 15 transport nodes, 2 UPFs, and 1 server. The slices were distributed equally, with two slices configured on each gNodeB and UPF instance. The monitoring, decision, and the network slice selection functionality are performed by a separate entity, which we will refer to as the engine. Currently, monitoring is only happening from the UEs' point of view. Based on the monitored KPI metrics, the fuzzy algorithm is evaluated, and the engine makes an appropriate decision and slice selection. The testbed is built on an OpenStack cluster [37], where we have created the necessary virtualized infrastructure. That includes the network, subnets, and network elements (manifested by OpenStack Virtual Machines (VM)). The UEs had 1 virtual Central Processing Unit (vCPU) with 2 GB of Random Access Memory (RAM), as well as the transport nodes and the server. The gNodeB-s,

UPFs used 2 vCPUs and 4 GB of RAM. The engine had different configurations, depending on whether it used a Mamdani or ANFIS-based fuzzy controller. The former had 2 vCPUs and 4 GB of RAM, while the latter had 32 vCPUs and 64 GB of RAM. This is because ANFIS training is an extensive task in itself, which requires a significant workload to manage these activities. All the network components were Ubuntu distros running Linux kernel 20.04. We used Crossplane [38] to automate infrastructure deployment. To implement our proposed Mamdani-based fuzzy algorithm, we used the scikit-fuzzy [39] Python library. For the ANFIS, we used a scikit-learn [40] compatible source code [31].

4.2 Measurement workflow

We used *iperf* [41] and *ping* [42] traffic measurement tools for performance evaluation. During the measurement latency, throughput and jitter were considered as KPIs. We measured throughput using *iperf* and latency using *ping*; jitter was calculated post hoc from the measured latency. The time intervals between measurements were set to 1 second for both ping and iperf. We continuously analyzed the KPIs from the UE's perspective, and the fuzzy-based algorithm was executed 20 times, with the inputs fetched from the log files. Always, the last 20 entries from the logs for each metric were processed, and the inter-slice handover decision was made based on the algorithm's outcome. We considered degraded KPI conditions for one measurement as follows: if the calculated crisp value exceeded a threshold, referred to as the *Per Call Threshold*. Additionally, we counted how many times it exceeded the *Per Call Threshold*. We checked at the end of the measurement whether the counted value exceeded the *Counter Threshold*. The final handover decision was made based on this counter. In fact, when we set this value, we control how rigorously we consider the SLA in certain measurement cases. Practically, we can set a desired SLA for the examined slice in this 20-second timeframe by this formula (3):

$$SLA[\%]=((1\text{-}Counter\ Threshold)/20)\times 100 \tag{3}$$

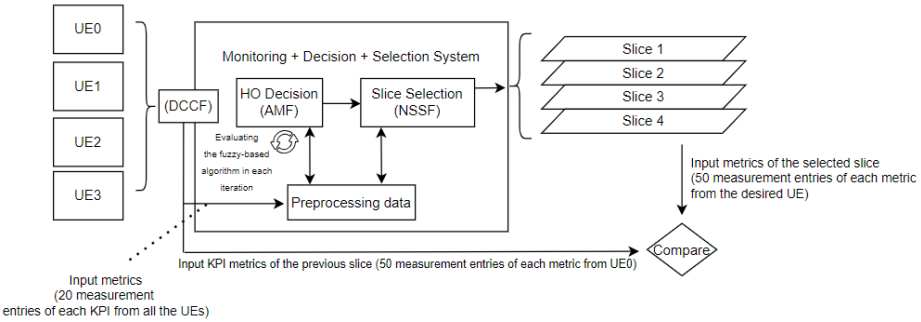

Fig. 9. Performance evaluation schematic

We can also calculate the actual SLA and compare it with the desired value, as we did during the measurements. Additionally, we can consider SLA in the context of the measured KPIs by examining the logs before and after the inter-slice handover. Of course, we need reference values for each KPI, and then we can derive the slices' SLAs by comparing each measure against the SLA-defined value. For now, we haven't conducted such an examination, as further consideration is needed regarding the selection of reference KPI values. Instead, we compared the average KPI values and calculated their proportions. We examined 50 log entries before and after the inter-slice handover of each metric. Based on that, we could derive the handover efficiency by comparing the average values. A schematic of the measurement procedure is shown in **Fig. 11**. We measured 4 test scenarios for each type of fuzzy algorithm. The main differences were the radio link setup and the membership function configuration.

The radio links were emulated using the Linux netem module [43]. The Linux traffic control (TC) [44] utility enables the configuration of the Linux kernel packet scheduler. This is achieved by using *qdiscs*, which define a specific packet scheduling mechanism. When using *netem*, you effectively configure a *qdisc* to tailor traffic behavior to your needs using netem-specific parameters. You can set the average delay, Jitter, packet loss, traffic distribution, and even the correlation of latency values, among other parameters. These parameters enable the configuration of various radio connectivity scenarios. In our testbed, we only emulated stationary UEs connected to the gNodeBs. This was achieved by parameterizing a constant average delay, Jitter, and setting the traffic distribution to normal.

4.3 Results on Mamdani-based Fuzzy logic for inter-slice handover management

Basically, across these 4 measurement scenarios, we were curious about how the efficiency of the inter-slice handover decision algorithms changes as we switch between these radio conditions. First, we set relatively different average delay and jitter values between the slices. As we moved further, we lowered the parameter differences. During

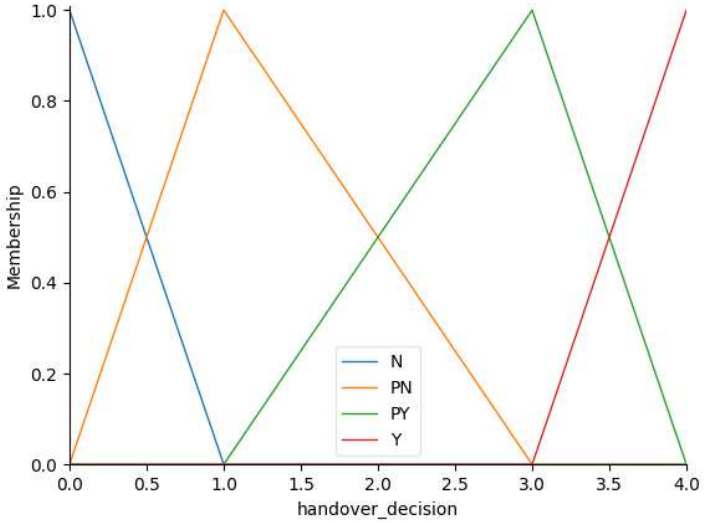

Fig. 10. Consequent variables configuration

the measurement, we always had 1 reference slice in which the examined UE was placed. Moreover, we had 3 alternative slices on hand to hand over the UE if needed. Deterministically, we always configured the remaining 3 slices with the same radio conditions. The configuration vectors are the following (Slice1_Slice2_Slice3_Slice4):

- 30-10_X_X_X
- 30-10_20-8_20-8_20-8
- 30-10_20-10_20-10_20-10

- 30-10_30-10_30-10_30-10

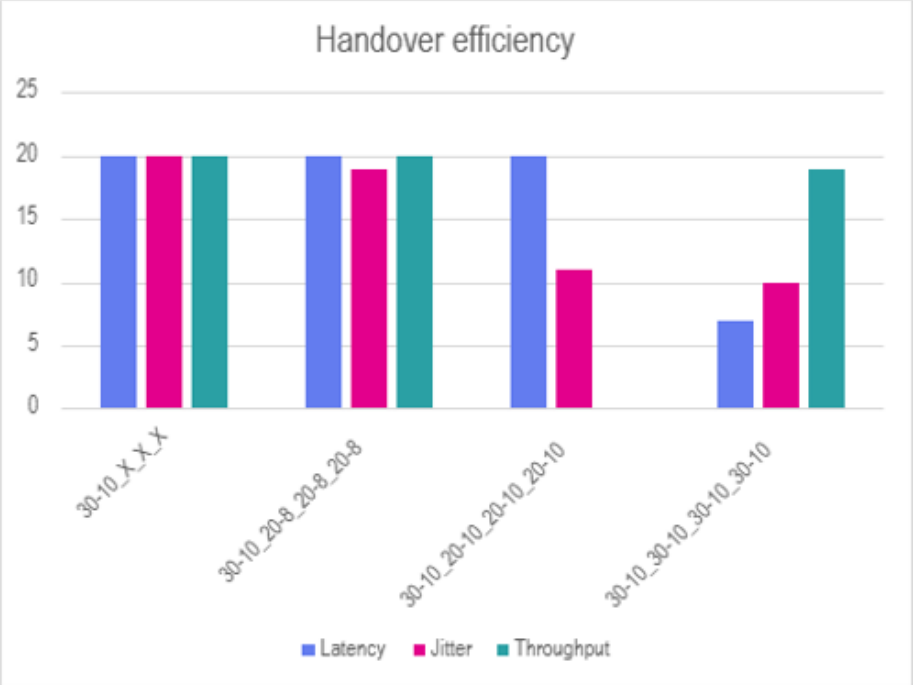

Fig. 11. Handover efficiency of the Mamdani-based algorithm

X means that we didn't configure a normal distribution on that radio interface. We used triangular membership functions with centroid defuzzification in all measurements. The consequent function configuration was set as shown in **Fig. 12**. The membership functions were fine-tuned for each measure based on the traffic conditions.

Based on **Fig. 13**, the algorithm performed well in the first two cases. Degradation in the KPI proportions is evident in the third and fourth measures. That's where the radio configuration parameters tended to be more similar across the referenced background slices.

4.4 Results on ANFIS-based inter-slice handover management

The same measurement series was conducted for our ANFIS-based scheme. As we discussed earlier, the ANFIS training is supervised; therefore, we needed to generate labels

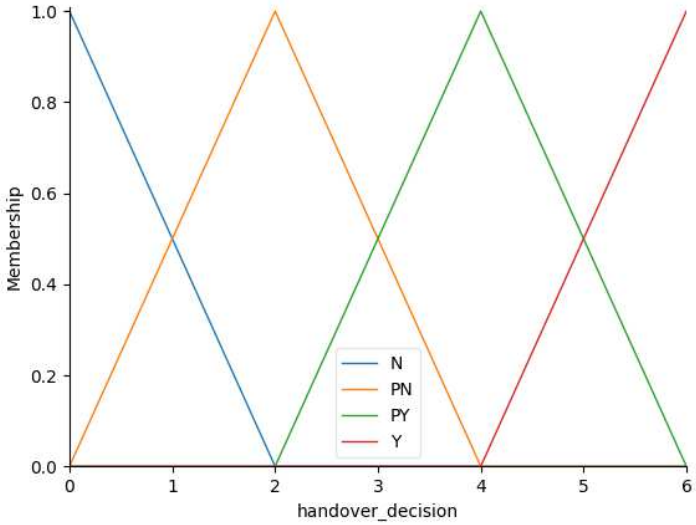

Fig. 12. Modified consequent variables for traffic generation

for the data points accordingly. In this case, it means the expected output on the scale that the algorithm would produce. For this, we utilized our Mamdani-based fuzzy algorithm to generate outputs for the relevant cases. We planned to create a generic inter-slice handover decision framework capable of recognizing all traffic patterns we examined in the previous "manual" fuzzy-based algorithm (30-10_X_X_X, 30-10_20-8_20-8_20-8, 30-10_20-10_20-10_20-10, and 30-10_30-10_30-10_30-10). That means all the generated datasets were mixed and fed to our ANFIS model. Each network traffic dataset corresponded to a specific radio configuration and comprised a 1-hour timeline of measurements. We haven't covered all possible configurations; we were only curious about specific scenarios:

- For the referenced slice, we used the combination of 20, 30 values (ms) for average delay, and 8, 10 (ms) for jitter. That realizes a 2x2 matrix.
- We always kept the values of the 3 remaining slices fixed and equal to each other. Additionally, we considered cases where we didn't put normally distributed traffic on the remaining slices.
- We modified the configuration of the consequent function for label generation to achieve performance enhancement (see **Fig. 14**).

Based on this, 2*2*4 (the first matrix multiplied by the 4 fixed, equal parameters on the other slices) + 4 cases (scenarios where we configure netem only for the reference slice) are covered, for a total of 20 cases. These measurements already cover the examined cases, and we can avoid feeding our model with superfluous scenarios (the total number of measurements would be (4*4)^4 = 65536).

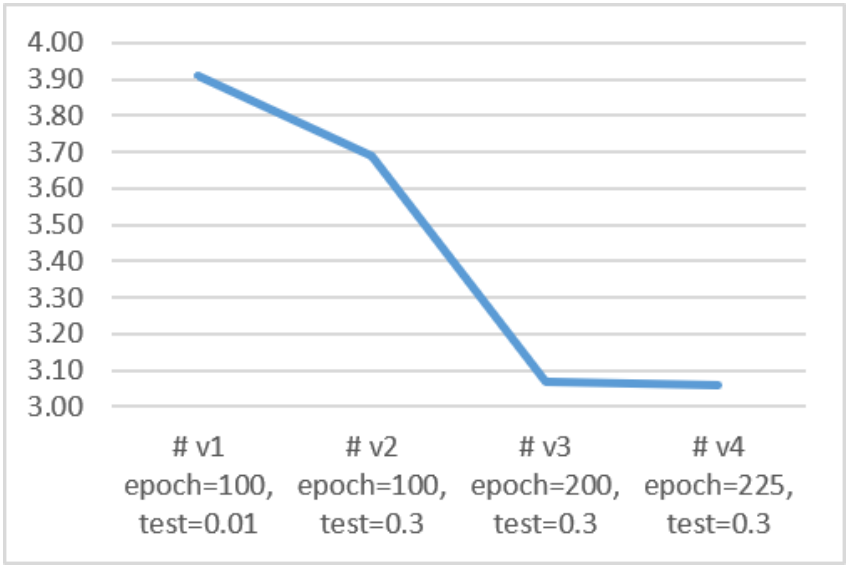

Fig. 13. Evolution of MSE during ANFIS training

The Calculated Mean Squared Error (MSE) was relatively high in all model training cases. Basically, two parameters were fine-tuned: the training data proportion (compared to the entire dataset) and the number of epochs. Through experimentation, we determined that the test size should be increased to 0.3 and the number of epochs should be set to approximately 225. This is the most optimal case we could take out of the system with these settings. Of course, subsequent functions can also be introduced for further examination. For future work, we will also consider that. For now, approaching the input-output relation is implemented entirely by this equation (4):

$$O4=wi'fi=wi'(Latency+Jitter+1/Throughput) \quad (4)$$

Here, the Latency, Jitter, and Throughput are the input variables. Based on **Fig. 15**, the optimized training configuration is still not an ideal choice as the MSE is around 3.06. That's quite high if we compare with the scale interval [0,6] (see in **Fig. 14**). Despite

this fact, it is still considerable for initial test measurements, because our use case is not a rigorous value regression but rather generating an output value that is fitted within the [Per Call Threshold, 6] interval. However, it still poses the question of how it affects the slice selection mechanism. Also, it's worth noting that with ANFIS, we use the entire large dataset to make handover decisions. That means we don't have to make unique adjustments to membership function parameters per measurement, because we have this pre-generated dataset as the source of truth. The same traffic pattern is observed in the first two measurements (**Fig. 16**), followed by a significant KPI degradation in the



Fig. 14. Handover efficiency of the ANFISs-based algorithm

third and fourth cases. Interestingly, some KPIs performed better in either Mamdani-based or ANFIS-based algorithms. Moreover, it's worth noting that in both cases, the throughput KPI had a massive degradation in the third measure. We observed the same phenomenon during dataset generation, where throughput decreased drastically in the alternative 3-slice case. That means, it is a reproducible radio condition that shouldn't be attributed to the performance of our inter-slice handover decision algorithms.

4.5 Performance Comparison

Based on the measures, if we reduce the parameter differences for the radio links across slices, both algorithms begin making inappropriate decisions. In the case of Mamdani, it performed better on Jitter in the second measure and on throughput in the fourth measure. However, the ANFIS-based decision algorithm performed better at selecting a slice with better Jitter results in the third case and better latency results in the fourth case. These are visualized in **Fig. 17**, where we verify our findings numerically. Basically, we just compare the handover efficiency metrics introduced in Section 4.2. We represent the metric differences per measurement in the figure. Positive values indicate better performance with Mamdani, while negative values indicate better performance with ANFIS. Zero value means equal performance results. One outstanding value is the throughput in the fourth measure, where our Mamdani-based interslice handover framework performs much better than the ANFIS-based one.

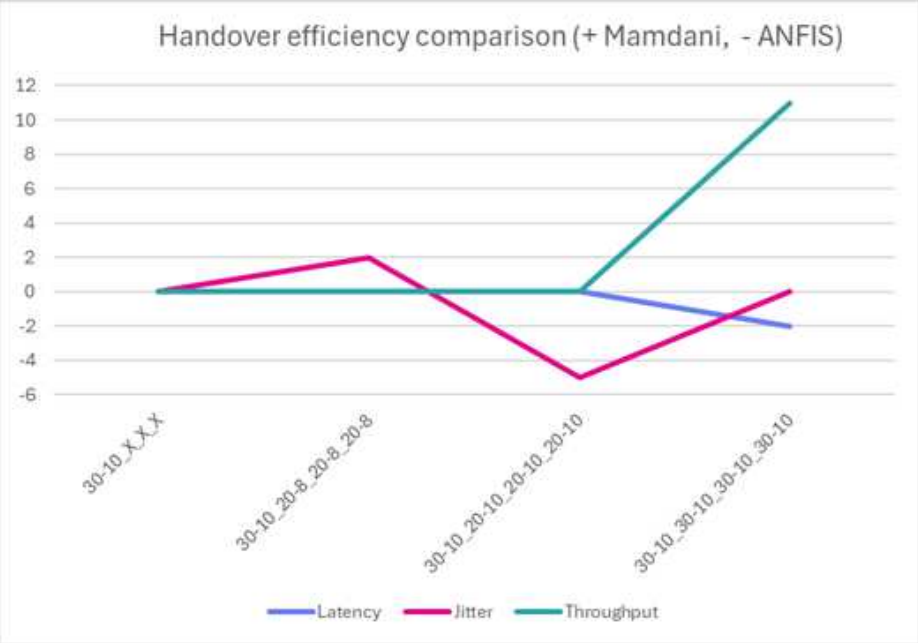

Fig. 15. Handover efficiency comparison

5 Conclusion and Future Work

This research represents a natural evolution of our earlier work, as published in [1][2]. The main architecture has matured sufficiently to begin KPI performance evaluations and test the functionality of our proposed fuzzy-based approaches. Also, a 3GPP-compliant framework architecture was created for this work, which aligns well with

standardization. Based on the performance evaluation, we conclude that our Mamdani- and ANFIS-based inter-slice handover management framework has potential. It is suitable for inter-slice handover decisions even with different radio configurations. Future research could make the framework more mature and agile for any test case. Of course, more measurements could be taken, and numerous test cases could be covered across SLA violation values, radio link configurations, and slice stability, with additional visualizations showing performance differences between slices after inter-slice handovers. Moreover, additional traffic data could be collected, as the size of our current dataset does not appear completely sufficient for ANFIS to be as efficient as, or at least better than, our Mamdani-based fuzzy algorithm across all KPI cases. Additionally, it remains an open research question whether we can optimize ANFIS to perform better across all KPIs. Moreover, it is worth considering whether ANFIS can promptly serve as a closed-loop orchestrator that handles all radio traffic conditions without manual fine-tuning of membership functions (in the case of Mamdani, this is the case). Regarding the labeling of teaching data, optimization of the Mamdani-based fuzzy engine, and the initial parameterization of ANFIS, a significant amount of fine-tuning is needed in the membership functions. Also, other functions could be considered than the triangle. Moreover, we have not even begun to address the rule base yet. It is also an interesting question whether it is possible to achieve better performance by revising the rule base. Not to mention that alternative defuzzification methods exist; currently, we use only the centroid method. More defuzzification strategies should be tested to see whether they achieve better performance. In the case of ANFIS, other consequent functions need to be examined to determine whether MSE performs better during ANFIS learning.

References

[1]. D. Matusovits, Á. Leiter, L. Bokor; "Towards a fuzzy-logic-based inter-slice handover management scheme in 3GPP 5G and beyond"; 3rd Workshop on Intelligent Infocommunication Networks, Systems and Services; pp. 93-98; February 2025; Budapest, Hungary, doi: 10.3311/WINS2025-016.

[2]. D. Matusovits, Á. Leiter, L. Bokor; "A testbed design to evaluate Fuzzy-based inter-slice handover solutions for 5G and beyond", IEEE 25th International Symposium on Computational Intelligence and Informatics (CINTI 2025), 18-20 November 2025, Budapest, Hungary, doi: 10.1109/CINTI67731.2025.11311745.

[3]. 3GPP TS 21.915 V0.2.0, "Release 15 Description; Summary of Rel-15 Work Items", Release 15, July 2018.

[4]. 3GPP TS 28.530 V18.1.0, "Management and orchestration; Concepts, use cases and requirements", Release 18, September 2024

[5]. 3GPP TS 28.531 V18.7.0, "Management and orchestration; Provisioning", Release 18, September, 2024

[6]. ETSI, GR NFV-EVE 012 V3.1.1, "Network Functions Virtualisation (NFV) Release 3; Evolution and Ecosystem; Report on Network Slicing Support with ETSI NFV Architecture Framework", December 2017

[7]. I. Stanic, D. Drajic and Z. Cica, "Overview of Network Selection and Vertical Handover Approaches and Simulation Tools in Heterogeneous Wireless Networks," 2023 16th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS), Nis, Serbia, 2023, pp. 133-142, doi: 10.1109/TELSIKS57806.2023.10315998.

[8]. A. Ahmed, L. M. Boulahia and D. Gaïti, "Enabling Vertical Handover Decisions in Heterogeneous Wireless Networks: A State-of-the-Art and A Classification," in IEEE Communications Surveys & Tutorials, vol. 16, no. 2, pp. 776 811, Second Quarter 2014, doi: 10.1109/SURV.2013.082713.00141.

[9]. Xiaohuan Yan, Y. Ahmet Şekercioğlu, Sathya Narayanan, A survey of vertical handover decision algorithms in Fourth Generation heterogeneous wireless networks, Computer Networks, Volume 54, Issue 11, 2010, Pages 1848-1863, ISSN 1389-1286, doi: 10.1016/j.comnet.2010.02.006.

[10]. Titu-Marius I. Băjenescu; "5G AND 6G EVOLUTION"; Journal of Engineering Science Vol. XXIX, no. 2(2022), pp. 68 - 75 Fascicle; 10 June 2022; Chișinău, Republic of Moldova, doi: 10.52326/jes.utm.2022.29(2).07.

[11]. W. Wu et al., "AI-Native Network Slicing for 6G Networks," in IEEE Wireless Communications, vol. 29, no. 1, pp. 96-103, February 2022, doi: 10.1109/MWC.001.2100338.

[12]. J. Mei, X. Wang and K. Zheng, "An intelligent self-sustained RAN slicing framework for diverse service provisioning in 5G-beyond and 6G networks," in Intelligent and Converged Networks, vol. 1, no. 3, pp. 281-294, Dec. 2020, doi: 10.23919/ICN.2020.0019.

[13]. Dangi, R., Lalwani, P. Optimizing network slicing in 6G networks through a hybrid deep learning strategy. J Supercomput 80, 20400–20420 (2024). doi: 10.1007/s11227-024-06238-y

[14]. M. A. Habibi et al., "Toward an Open, Intelligent, and End-to-End Architectural Framework for Network Slicing in 6G Communication Systems," in IEEE Open Journal of the Communications Society, vol. 4, pp. 1615-1658, 2023, doi: 10.1109/OJCOMS.2023.3294445.

[15]. A. Ali and H. Arslan, "AI-Driven Digital Twins: Optimizing 5G/6G Network Slicing With NTNs," in IEEE Wireless Communications Letters, vol. 15, pp. 66-70, 2026, doi: 10.1109/LWC.2025.3618622.

[16]. R. Pang, X. Zhang, C. Ling, M. Wang, Z. Shao and Y. Wang, "AI-Native Latency Optimization for 6G Industrial IoT: An Interpretable Machine Learning Approach to Robot Control Network Slicing," 2025 5th International Conference on Artificial Intelligence, Robotics, and Communication (ICAIRC), Xiamen, China, 2025, pp. 818-822, doi: 10.1109/ICAIRC68035.2025.11384866.

[17]. Botez, R.; Zinca, D.; Dobrota, V. Redefining 6G Network Slicing: AI-Driven Solutions for Future Use Cases. Electronics 2025, 14, 368. doi: 10.3390/electronics14020368.

[18]. M. Yaqoob, R. Trestian, M. Tatipamula and H. X. Nguyen, "Digital-Twin-Driven End-to-End Network Slicing Toward 6G," in IEEE Internet Computing, vol. 28, no. 2, pp. 47-55, March-April 2024, doi: 10.1109/MIC.2023.3332252.

[19]. S. Kukliński, L. Tomaszewski, R. Kołakowski and P. Chemouil, "6G-LEGO: A framework for 6G network slices," in Journal of Communications and Networks, vol. 23, no. 6, pp. 442-453, Dec. 2021, doi: 10.23919/JCN.2021.000025.

[20]. M. Başaran, S. Aktaş and B. Bilgin, "6G Network Slicing Vision for Post-Disaster: AI-Enabled User Prioritization and Energy Management," 2024 IEEE International Conference on Advanced Telecommunication and Networking Technologies (ATNT), Johor Bahru, Malaysia, 2024, pp. 1-4, doi: 10.1109/ATNT61688.2024.10719190.

[21]. A. M. Escolar, J. B. Bernabe, J. M. A. Calero, Q. Wang and A. Skarmeta, "Network slicing as 6G security mechanism to mitigate cyber-attacks: the RIGOUROUS approach," 2024 IEEE 10th International Conference on Network Softwarization (NetSoft), Saint Louis, MO, USA, 2024, pp. 387-392, doi: 10.1109/NetSoft60951.2024.10588887.

[22]. C. You, X. He, J. Xu, P. Yang and T. Q. S. Quek, "Sustainable Service-Oriented RAN Slicing for AI-Native 6G Networks," 2023 21st International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), Singapore, Singapore, 2023, pp. 1-5, doi: 10.23919/WiOpt58741.2023.10349874.

[23]. R. M. A. Molina, H. Assem Alameddine, M. Pourzandi and C. Assi, "PUL-Inter-Slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks," in IEEE Transactions on Information Forensics and Security, vol. 20, pp. 12887-12902, 2025, doi: 10.1109/TIFS.2025.3640828.

[24]. Chrysafiadi, K. (2023). Fuzzy Logic-Based Software Systems. In: Fuzzy Logic-Based Software Systems. Learning and Analytics in Intelligent Systems, vol 34. Springer, Cham. doi: 10.1007/978-3-031-44457-9_3.

[25]. Ampririt, P., Qafzezi, E., Bylykbashi, K., Ikeda, M., Matsuo, K., Barolli, L. (2022). A Fuzzy-Based System for Handover in 5G Wireless Networks Considering Different Network Slicing Constraints: Effects of Slice Load Parameter on Handover Decision. In: Barolli, L., Miwa, H., Enokido, T. (eds) Advances in Network-Based Information Systems. NBiS 2022. Lecture Notes in Networks and Systems, vol 526. Springer, Cham. doi: 10.1007/978-3-031-14314-4_15.

[26]. Ampririt, P., Qafzezi, E., Bylykbashi, K., Ikeda, M., Matsuo, K., Barolli, L. (2022). A Fuzzy-Based System for Handover in 5G Wireless Networks Considering Network Slicing Constraints. In: Barolli, L. (eds) Complex, Intelligent and Software Intensive Systems. CISIS 2022. Lecture Notes in Networks and Systems, vol 497. Springer, doi: 10.1007/978-3-031-08812-4_18.

[27]. Ampririt, P., Higashi, S., Qafzezi, E., Ikeda, M., Matsuo, K., Barolli, L. (2024). A Fuzzy-Based System for Selection of Slice Service Type in 5G/6G Wireless Networks. In: Barolli, L. (eds) Advances in Network-Based Information Systems. NBiS 2024. Lecture Notes on Data Engineering and Communications Technologies, vol 224. doi: 10.1007/978-3-031-72325-4_2.

[28]. Ampririt, P., Higashi, S., Qafzezi, E., Ikeda, M., Matsuo, K., Barolli, L. (2025). A Fuzzy-Based System for Handover in 5G/B5G Wireless Networks: Effect of Slice Active Time on Handover Decision. In: Barolli, L. (eds) Advances on Broad Band Wireless Computing, Communication and Applications. BWCCA 2024. Lecture Notes on Data Engineering and Communications Technologies, vol 231. Springer, Cham. doi: 10.1007/978-3-031-76452-3_3.

[29]. Phudit Ampririt, Shunya Higashi, Ermioni Qafzezi, Makoto Ikeda, Keita Matsuo, Leonard Barolli, An intelligent fuzzy-based system for handover decision in 5G IoT networks considering network slicing and SDN technologies, Internet of Things, Volume 23, 2023,100870, ISSN 2542-6605, doi: 10.1016/j.iot.2023.100870.

[30]. Komaragiri Srinivasa Raju, Dasika Nagesh Kumar, "Artificial Intelligence and Machine Learning Techniques in Engineering and Management", Springer Singapore, 2025, doi: 10.1007/978-981-96-2621-2

[31]. Zhang, D., Chen, T. Scikit-ANFIS: A Scikit-Learn Compatible Python Implementation for Adaptive Neuro-Fuzzy Inference System. Int. J. Fuzzy Syst. 26, 2039–2057 (2024). doi: 10.1007/s40815-024-01697-0

[32]. Giannini, F., Diligenti, M., Maggini, M. et al. T-norms driven loss functions for machine learning. Appl Intell 53, 18775–18789 (2023). doi: 10.1007/s10489-022-04383-6

[33]. Montesinos López, O.A., Montesinos López, A., Crossa, J. (2022). Fundamentals of Artificial Neural Networks and Deep Learning. In: Multivariate Statistical Machine Learning Methods for Genomic Prediction. Springer, Cham. doi: 10.1007/978-3-030-89010-0_10

[34]. Tomasiello, S., Pedrycz, W., Loia, V. (2022). Fuzzy Inference Systems. In: Contemporary Fuzzy Logic. Big and Integrated Artificial Intelligence, vol 1. Springer, Cham. doi: 10.1007/978-3-030-98974-3_5

[35]. J. . -S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," in IEEE Transactions on Systems, Man, and Cybernetics, vol. 23, no. 3, pp. 665-685, May-June 1993, doi: 10.1109/21.256541

[36]. 3GPP TS 23.288 V19.4.0, "Architecture enhancements for 5G System (5GS) to support network data analytics services", Release 19, September 2025

[37]. "OpenStack documentation". Accessed: March. 28, 2026. [Online]. Available: https://docs.openstack.org/2025.2/

[38]. "Crossplane documentation". Accessed: March. 29, 2026. [Online]. Available: https://docs.crossplane.io/latest/

[39]. "Scikit-Fuzzy Python Library". Accessed: Aug 2, 2026. [Online]. Available: https://github.com/scikit-fuzzy/scikit-fuzzy

[40]. "Scikit-Fuzzy Python Library". Accessed: Aug 2, 2026. [Online]. Available: https://scikit-learn.org/stable/

[41]. "Iperf documentation". Accessed: March 29, 2026. [Online]. Available: https://iperf.fr/iperf-doc.php

[42]. "Ping documentation". Accessed: March 29, 2026. [Online]. Available:https://man7.org/linux/man-pages/man8/ping.8.html

[43]. "TC - netem module". Accessed: Aug. 2, 2026. [Online]. Available: https://man7.org/linux/man-pages/man8/tc-netem.8.html

[44]. "TC Manual". Accessed: Aug. 2, 2026. [Online]. Available: https://man7.org/linux/man-pages/man8/tc.8.html