This is the author's version of the article. The final version published by IEEE is A. Collet, A. Bazco-Nogueras, A. Banchs and M. Fiore, "Learning to Learn How to Manage Network Resources with Loss Function Meta-Learning," in IEEE Communications Magazine, accepted to be published, 2024


# Learning to Learn How to Manage Network Resources with Loss Function Meta-Learning

Alan Collet, *Student Member, IEEE*, Antonio Bazco-Nogueras, *Member, IEEE*, Albert Banchs, *Senior Member, IEEE*, and Marco Fiore, *Senior Member, IEEE*

***Abstract*—The evolution of communication networks towards self-configuring systems requires the development of anticipatory approaches for network management to realize the envisioned concept of a zero-touch network orchestration. Current anticipatory network intelligence solutions rely on a well-defined loss function, which means that they require perfect knowledge of the relationship between the proactive management decisions and the consequent system performance. However, in anticipatory networking, there exist many tasks where characterizing such a relationship in advance is not possible. In such tasks, it is possible to measure the resulting performance of a management decision *a posteriori*, but we cannot know *a priori* the resulting performance of a certain management decision. A simple example of such tasks could be the maximization of the monetary profit when allocating resources to end users: when taking a certain resource allocation decision, it is possible to measure the profit afterwards, but it would be extremely difficult to determine a priori the resulting monetary profit. To close this gap, we present a novel two-fold learning approach, which is able to jointly learn the relationship between the prediction and the target management objective at the same time as it apprehends to anticipate the corresponding task. This method lays the foundations to the automated adaptation of network intelligence to specific complex objectives in zero-touch network management. We apply this method to different use cases of interest including monetary profit maximization.**

***Index Terms*—Zero-touch networking, loss learning, anticipatory network management**

## I. Introduction

COMMUNICATION networks are experiencing a considerable increment in operation complexity to accommodate the growing heterogeneity of service requirements, which strains slow, reactive, human-in-the-loop techniques currently adopted in network management. This calls for network automation based on Machine Learning (ML) solutions [1] that support Intent-Based Networking (IBN) and Zero-touch Network and Service Management (ZSM) [2]. Dedicated architectural models are being proposed to enable the co-existence of ML Operations (MLOps) frameworks with traditional network operations [3], paving the road to an effective ML-based Network Intelligence (NI) that achieves the goals above.

### A. Network Intelligence for Anticipatory Resource Management

Among the functionalities that NI can automate, anticipatory network resource management plays a key role, as it promises to cut down operator's capital and operating expenses (OPEX) without affecting the Quality of Experience (QoE) of users.

Alan Collet and Albert Banchs are with IMDEA Networks Institute, and Universidad Carlos III de Madrid. Antonio Bazco-Nogueras and Marco Fiore are with IMDEA Networks Institute.

While anticipatory network management inherently builds on predictions, standard forecasting techniques are not directly applicable to network management tasks, which are characterized by unique relationships between the decisions and the network performance. Relying on legacy predictors creates a mismatch between the *loss*, i.e., the objective of the forecast, and the *metric*, i.e., the network management performance target, which is a well-known problem in machine learning [4]. The issue is addressed in two ways by current works in NI for network management. A first approach (cf. [5]) is that of running a standard forecasting model of the traffic demand, and leveraging the prediction to manually decide on suitable network configuration updates. The second (cf. [6]) relies on expert-designed loss functions that map the relationship between future variables and network performance.

### B. Limitations of Current Approaches in Practical Use Cases

While the two strategies above may provide excellent results in specific problems, they suffer from an inherent limitation: the need for manual tailoring to each single task. The drawback is twofold: $(i)$ it requires a non-scalable amount of human expertise and time, and $(ii)$ it assumes that the objective function can be fully characterized at the moment of design. Point $(i)$ above breaks the full automation of the system, while assumption $(ii)$ does not hold in many networking tasks, including those listed next — where the objective function is not known at design time yet it can be measured (hence learned) at run time.

**Carbon footprint minimization.** Optimizing energy consumption is a crucial objective for operators. However, obtaining a perfect a priori knowledge of how resource-allocation decisions affect the actual energy consumption of the infrastructure is a daunting task. For instance, let us consider a virtualized Radio Access Network scenario, where the functional splits between large sets of Distributed Units (DUs) and Central Units (CUs) must be decided. The responsible network function would be oblivious to aspects like the policy for job parallelization and core management in the datacenter hosting the CU, or its exact power consumption behavior. Since the decisions for the DUs served by the same CU impact each other, selecting the functional splits that minimize the energy footprint is an example of intertwined network predictions under an objective that is unknown a priori but can be measured a posteriori.

**QoE and revenue optimization.** In a network slicing scenario where the operator must ensure the QoE of end users by configuring slice resource reservation, a full characterization

of how the network management decisions impact the revenues is not possible, since ($i$) the operator may observe coarse QoE degradation via QoE monitoring tools, yet has no direct visibility to the end-user QoE measurements that are only available to the service provider (SP), and ($ii$) the decisions at each section of the network and for each user create an entangled system that is difficult to model.

**Anomaly detection and network healing** requires NI algorithms, e.g., to trigger the correct remote unit configurations to mitigate the effect of an upcoming abnormal demand. However, knowing in advance how specific antenna configurations affect the end-user performance, and how false positives/negatives in the anomaly detection influence the cost for the operator (e.g., in terms of churn of subscribers) is not feasible in general.

### C. Making Anticipatory Network Management Zero-Touch

We argue that the inherent problem underlying current state-of-the-art approaches to automated anticipatory networking is that they build on an inflexible NI design that only targets a single objective and is not applicable to other (even similar) problems. To overcome this limitation, we present an original forecasting paradigm for fully automated NI, i.e., *loss meta-learning*. Our proposed approach self-learns unknown and complex relationships between the management decision and the network performance, hence it *learns to predict* to optimize complex objectives and not just to reduce the prediction error.

From a practical viewpoint, the loss meta-learning NI is implemented as jointly trained dual neural networks (NNs), with ($i$) a task regressor block in charge of producing the actual (anticipatory) network management decision, and ($ii$) a loss-learning block, which learns the decision–performance relationship while training the regressor. This design is able to learn differentiable approximations of loss functions (i.e., performance metrics) that are correlated in time, non-differentiable, and/or may depend on several independent variables, and it outperforms current solutions in terms of performance while also providing the extra degree of automation. We recently proposed some specific implementations [7], [8], yet the idea of self-learning unknown loss functions applies to many practical problems as the ones described before, where the performance measure deriving from the operator decisions cannot be obtained neither analytically nor manually because of its complexity or because it is not known a priori. We thus present in this paper the general framework, as well as new practical application use cases.

## II. State of the Art and Main Challenges in Anticipatory Networking With Complex Objectives

In network management, forecasting is usually not the main goal but an intermediate step to solve some optimization or control problem that requires future knowledge to be efficiently solved. The challenge of predicting the relevant future Key Performance Indicator (KPI) — e.g., traffic demand or network delay — has been thoughtfully analyzed in the literature, and current models are able to provide very accurate predictions [9]. Yet, there exists another challenge that has been generally overlooked: how to exploit those predictions to optimize the objective network performance metric; that is, the loss–metric mismatch. Recent works have shown the huge potential improvement that solving the loss–metric mismatch can bring [6]; however, this improvement has been limited so far to very specific management tasks.

Next, we describe the state of the art in anticipatory networking, which generally follows two strategies: ($i$) disentangling forecasting from the decision-making task, hence ignoring the possibly convoluted relationship between predictions and final objective metrics; or, ($ii$) considering that a perfect knowledge of that relationship is available and building the model design on it. Then, we explain the limitations of current solutions and the requirements needed to overcome such limitations in anticipatory networking. The different approaches, together with the proposed solution that will be later presented in the following section, are summarized in Figure 1, which portrays the key components of each NI strategy and how they fit the overall network ecosystem. There, we differentiate the ML-based elements from the hand-made expert-designed elements, and we also distinguish the workflow depending on whether the network is training the ML algorithms or it is operating the trained algorithms, such that we adjust to current MLOps and NI frameworks [3].

### A. State of the Art in Forecasting for Decision Making

The most direct approach to design NI for anticipatory networking is to apply standard forecasting methods to predict the future values of the network KPI relevant to the management decision; the result is then fed to some decision-making block that is manually developed based on expert knowledge.

These predictors were traditionally based on statistical models, Markovian theory or information theory [10], [11], but lately data-driven ML algorithms have become the state-of-the-art approaches for forecasting [12]. The generic forecasting models are trained to minimize generic error metrics of KPI values, such as the Mean Absolute Error (MAE) or Mean Squared Error (MSE), and they are oblivious to the operator's objective and therefore to the relationship between the output and the performance objective.

This strategy is illustrated in Figure 1.$a$. The forecast is blind to the management objective, and hence it cannot provide a solution for automated management decisions. Instead, after the forecast has been computed, the management decision-making is run by a different network element, which is different for each possible networking task as the performance depends on the said prediction in distinct manners.

### B. State of the Art in Loss Function Tailoring

While the previous approach treats prediction and decision-making as fully independent tasks, there are clear benefits in designing algorithms that jointly handle both aspects of the problem. First, knowing the possible heterogeneity of the forecast accuracy for different input ranges can improve the decision-making process by incorporating such variable uncertainty in the problem. Moreover, the average-minimum-error forecast method may not be optimal, as the decision

making can be very sensible to errors in a specific small input range, while being more robust at other ranges.

Some works have proposed ML-based models with tailored loss functions that drive the model parametrization during training [6], [13]. The customization of the loss seeks to emulate the relationship between predictions and the management objective. This approach also allows us to directly output the preemptive management decision from the past values input into the system. We illustrate this approach in case $b$) of Figure 1. This approach overcomes the main limitation seen in Section II-A: it tackles the loss–metric mismatch by having human experts design losses that are tailored to the actual system metric [4], and it has been shown to yield significant advantages in practical settings [6].

### C. Missing Component to Realize Automated Anticipatory NI

We next describe the main limitations of the state-of-the-art solutions, which are intrinsic to all existing anticipatory MANO models, regardless of whether they compartmentalize prediction and decision stages or integrate them via customized loss functions, and do not depend on the specific implementation.

The aforementioned solutions implicitly assume that we are able to manually characterize the underlying relationship between the available data, the taken decisions, and the performance objective. Yet, this may not be the case for many practical tasks, as exemplified in Section I-B.

Even when the objective function is known, the required ad hoc design for each of the countless management problems that might exist in practice entails an inherent limitation, and it is a critical barrier to one of the main requirements to ZSM and IBN: the capability to adapt to any problem by conforming the output to the management objective, and doing so automatically with minimum (if any) human intervention.

A very suitable paradigm to address the limitation above for anticipatory network management is *loss function meta-learning*. Meta-learning refers to the ability of *learning to learn*: not only training an agent into learning a given task, but also learning how to train the said agent; or, in other words, being capable of learning the complex relationship governing the objective performance even when that mapping may not be available at design time. For example, loss meta-learning is paramount when the decision depends on third-party information (e.g., by SPs) unavailable to the operator, or when the objective depends on the software and hardware equipment employed over the whole infrastructure.

The literature on meta-learning in the ML domain has focused on automatizing previously manual tasks for the development and tuning of ML models, such as cross-validation, hyper-parameter selection, or data cleansing. The application of meta-learning to loss function customization, which we propose in this paper, is instead still in its infancy, especially for regression: prior works only considered customizable loss functions for classification (whereas anticipatory network management relies on regression) based on a limited set of predefined expressions and parameters (which require manual design and limit the flexibility of the result) [14].

Ultimately, no existing approach fully aligns with the learning requirements set forth by a fully automated anticipatory network management, which calls for apprehending a *clean-slate loss*, without any prerequisite and assumption on how the loss function is except from the input and output dimensions. Next, we present our proposed strategy, which is to the best of our knowledge the first clean-slate loss meta-learning method for networking tasks, and which overcomes the aforementioned limitations of current approaches.

## III. Architecture, Advantages, and Novelty of Loss Meta-Learning for Anticipatory Networking

We propose a novel loss-learning architecture that automatically learns the complex relationship between network states and the actions to be taken by the network operator. It consists of a regressor that is a priori agnostic to the loss function and can adapt to different problems. It performs a bifold learning, since it simultaneously learns ($i$) the loss function that governs the particular use case, i.e., the *relationship*, and ($ii$) how to operate the target network functions or components, i.e., the *actions*, so as to minimize such a loss.

We remark that the proposed methodology is intentionally generic, as it applies to *any* network management task that comprises a prediction and a decision based on such a prediction. The design abstracts the internal architecture of the regressor and loss-learning blocks, leaving their implementation adaptable to the specifics of the target problem.

The distinctive characteristic of this architecture is that it is composed of two main blocks, as we can see in case $c$) of Figure 1. First, the *regressor* block is in charge of learning and executing the standard anticipatory management decisions, akin to the analogous "joint prediction and decision making" block in Figure 1.$b$. Second, there is a *performance loss-learning* block; this *loss-learning* block receives as inputs the decision of the *regressor* block (the decision taken) and the true network state, and it aims at learning the relationship between the performance objective and the management decisions taken by the first block; that is, the a priori unknown loss function. For that, it leverages historic data from a posteriori system measurements and network monitoring functions, e.g., taking advantage of the Management Data Analytics Function (MDAF) standardized by 3GPP Release 16.

The loss-learning block acts as the loss function to train the regressor block. This loss function evolves and gets refined during training to steer the system parameters toward optimizing the actual objective system performance. Once trained, the loss-learning block acts as a tailored loss function as in the approach presented in Section II-B and Figure 1.b, with the advantage that its design is now fully automated and cuts significant (time and monetary) human costs.

It is also worth noting that we do not train the system to minimize directly a certain legacy error metric between the regressor's actions and the ideal best action. Instead, as depicted in Figure 1.c, the training of the loss-learning block aims at minimizing a legacy error metric (e.g., MSE or MAE) between the output of the loss-learning block and the performance objective (for example, a specific KPI of interest). Hence, we train based on a legacy error metric — oval in Figure 1.c — on the estimated versus monitored performance. This implies that we do not require any mathematical modeling of the objective.

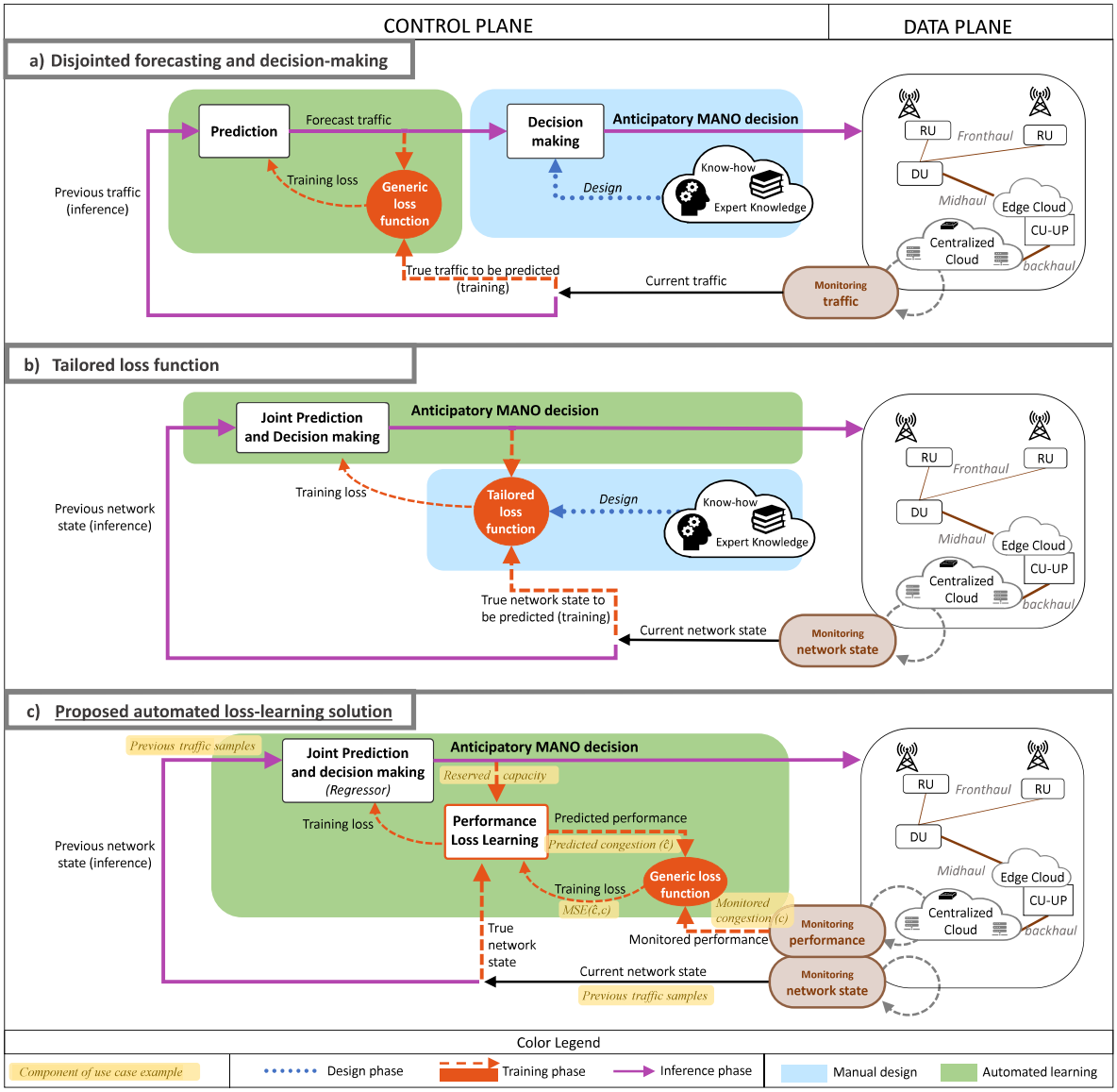


Fig. 1. State of the Art *a*) and *b*) and proposed solution *c*) for anticipatory Management and Network Orchestration (MANO) with complex objective functions. The proposed solution scheme in *c*) also shows a simple use case of capacity reservation for slices based on previous traffic volume to minimize congestion.

We provide a toy use case in Figure 1.c for clarifying the role of each part of the approach. This example consists of reserving the optimal capacity for each network slice for the next time interval at different sections of the network. The decision is based on previous traffic samples. The objective is to reduce congestion across the network. The relationship between reserved capacity and congestion cannot be analytically modelled. Thus, we use loss meta-learning to train the model *without knowing a priori the loss function.*

The specific architecture of each block (e.g., the depth of the NNs, or the type of layers) can be adapted to the problem's complexity or the available computing resources. A key property of our proposed design is that, when both blocks are implemented by means of deep NNs, they are simultaneously trained through the same gradient descent process, which speeds up convergence and ensures consistency between predictive action and performance objective. This approach can naturally incorporate state-of-the-art methods from AutoML to increase the model's adaptability and reduce human intervention, e.g., by embedding adaptive learning rate algorithms, or calibrating an exploration-exploitation trade-off similar to what is done in Reinforcement Learning.

Our proposed design deals with the inherent limitations of the existing approaches:

- We are able to learn, without human intervention and from measurement directly, differentiable representations of relationships between the management decisions and the performance objective that are unknown a priori, tangled, non-linear, non-differentiable and/or multivariate.
- We provide a generic framework that can solve many management tasks with minimal modifications, thus also removing the hindrance of re-designing each anticipatory NI solution on a refined case-by-case basis.
- The proposed design yields desirable features in terms

of *transfer learning* (as the trained loss-learning block can be reused across different scenarios, as shown in [8]), *explainability* (as, even if it does not constitute the solution explainable, the learned loss function can be inspected so as to interpret the network management choices), and *generalizability* (as it handles both competitive and cooperative behaviors), which are not available to, e.g., approaches based on Reinforcement Learning.

In conclusion, loss meta-learning paves the way towards actual network management automation, by streamlining the design of loss functions — an emerging paradigm that is still largely unexplored even in the machine learning community.

## IV. Performance in Sample Practical Use Cases

We evaluate the proposed loss meta-learning concept in two network management problems where the relationship between the predicted action values and the target performance are convoluted and partially or fully unknown a priori.

The input data corresponds to traffic volume measurements (mean throughput every five minutes) recorded in a commercial nation-wide mobile network for popular streaming services over two consecutive months. Both main blocks are implemented as small size NNs. Simulations run on a Python framework[1] based on TensorFlow and powered by one Nvidia A100 GPU. In the worst case among all the experiments performed, the training time of the proposed method is $2250 \pm 10$s, which is just slightly above the time required when both NNs are separately trained ($1430s + 620s = 2050s$). While the absolute time naturally depends on the complexity and the size of the NNs, we observe that this training time is comparable with standard methods.

### A. Resource Allocation to Individual Services

*1) Problem statement:* The network operator aims at proactively allocating the network resources (e.g., a guaranteed transport capacity in Mbps — the output decision) required to serve the demand of a certain service (e.g., YouTube) during the next 5 minutes, based on the observed demand for that service in the previous hour (e.g., 12 values representing the mean traffic generated by YouTube at every 5 minutes — the input data). The operator has a Service Level Agreement (SLA) with the SP, by which it receives a revenue if the service demand is fully served but to whom it has to pay a fee otherwise (resulting in an economic gain or loss — the performance metric to optimize). In this scenario, standard problem-agnostic predictions perform poorly [7] due to the intrinsic asymmetry of the problem: while overdimensioned resources incur a small linearly proportional cost, underprovisioning resources would lead to a SLA violation, in turn triggering a loss of operator's revenue. Hence, the economic cost is intrinsically different when the prediction error is positive or negative. This cost metric is represented by the gray continuous line in Figure 3.

We compare the proposed strategy against DeepCog, a state-of-the-art NN model using an expert-designed $\alpha$-OMC loss

[1]The code is available at: https://github.com/nds-group/AutoManager.

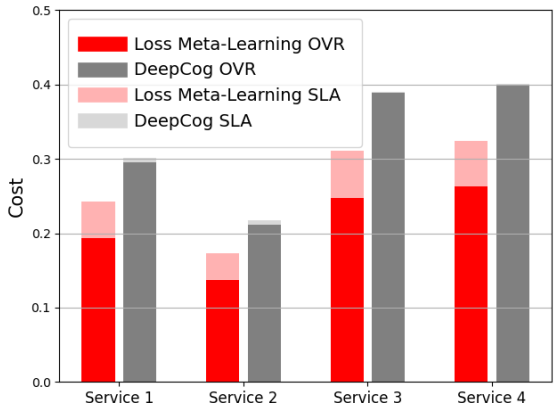


Fig. 2. Normalized performance cost of the anticipatory resource allocation. Left bars refer to our loss meta-learning solution, right bars refer to DeepCog [6]. Cost is broken down in SLA violations and overhead (OVR).

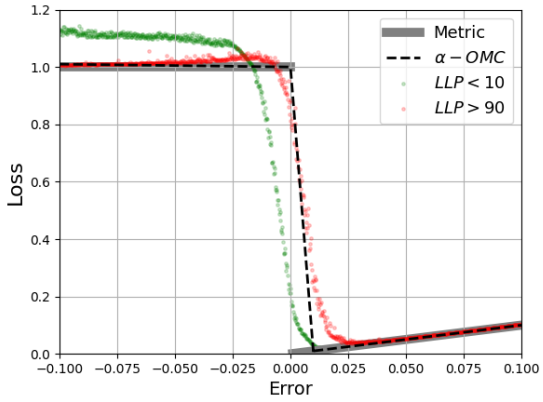


Fig. 3. Normalized performance cost (loss function) for our first use case: ($i$) ground-truth metric defining the relationship between decisions and performance (gray), ($ii$) expert-designed loss function that models the metric in DeepCog ( -OMC, dashed black), and ($iii$) fully automated loss meta-learning solution (LLP). For LLP, we show the learnt function for the top (red) and bottom (green) deciles of the predicted values, i.e., in presence of high and low traffic demands, respectively.

function *tailored* to this specific scenario [6]. The expert-knowledge-based loss function acts as a differentiable approximation to the asymmetric behavior described above.

*2) Results:* The detailed implementation is described in Table **??**. Long Short-Term Memory (LSTM) layers effectively capture and retain dependencies in sequential data.

Figure 2 summarizes the performance of loss meta-learning, which outperforms DeepCog in four settings characterized by different services. It does so by trading off the unnecessarily high overprovisioning induced by DeepCog with minimum SLA violations, reducing operator's overall costs.

The key to the improved performance of the loss meta-learning solutions is unveiled in Figure 3. There, the manual $\alpha$-OMC loss is designed as a bi-dimensional function of the error between the predicted capacity and the actual capacity required by the service (i.e., the abscissa values), which indeed mimics the ground-truth metric. Instead, our proposed approach (automatically) learns a more complex three-dimensional function over the complete space of all combinations of predicted and needed capacity. This property

TABLE I
IMPLEMENTATION OF REGRESSOR AND LOSS-LEARNING BLOCKS FOR THE USE CASES OF SECTION IV. ALL THE ACTIVATION FUNCTIONS ARE RELU.

| | | |
|---|---|---|
| Regressor | Use Case I | 3 LSTM $(128, 128, 64)$ |
| | Use Case II | 3 LSTM $(128, 128, 64)$<br>3 LSTM $(128, 128, 64)$<br>3 LSTM $(128, 128, 64)$<br>3 LSTM $(128, 128, 64)$<br>→ 4 Dense $\begin{pmatrix}256, 256\\128, 64\end{pmatrix}$ |
| Loss-learning Block | Use Case I | 3 Dense $(256, 256, 128)$ |
| | Use Case II | 5 Dense $(256, 256, 256, 256, 128)$ |

allows adapting the learned loss to the sensitivity of the predictor to specific system conditions, as illustrated in Figure 3, where, for the sake of clarity, we present a simplified view of the meta-learned loss for high and low demand volumes only. Our model finds different adapted curves to the two cases: in presence of high traffic loads it shifts the minimum loss point to the right to offer a safer margin against the higher errors incurred by the predictor in such traffic conditions.

### B. QoE-driven Admission Control and Resource Reservation

*1) Problem statement:* We consider the management task of anticipatory admission control (AC) and resource reservation (RR) for network slices. The operator uses information about the past demands associated to every slice (input) to take decisions on which slices shall be accepted and how many resources reserved for each accepted slice at every 5 minutes (output). The goal is maximizing revenues originating from admitted slices, while ensuring the QoE levels defined in the associated SLAs (performance metric). More precisely, if a slice is accepted, the SP pays to the operator a fee proportional to some KPIs as indicated in the SLA (e.g., volume of serviced traffic); yet, this is subject to the achievement of a certain QoE level, otherwise the SLA is violated and the operator incurs into a penalty fee.

The operator can protect from the risk above by over-allocating resources to admitted slices; however, this may lead to the exclusion of additional slices and their associated revenues, and the unnecessary resources generate extra costs because OPEX are proportional to the reserved resources.

There are several aspects that make it difficult for the operator to know the relationship between AC-RR decisions and revenue before the decisions are actually enacted in the production system. First, such a relationship is a complex function that depends on many operational parameters and application-level data (e.g., user feedback on QoE) that the operator does not have access to. Second, the decision for each SP is entangled, and accepting or not one SP impacts on the QoE of all SPs. This makes it impossible to manually conceive a loss function that captures the complex and partially unknown relationship. While QoE is a subjective metric, there exist methods that model and quantize it. The comprehensive model that relates network management decisions and QoE is detailed in [8, V.A].

TABLE II
AVST FOR THE FOUR SERVICES IN THE QOE-DRIVEN AC-RR USE CASE.

| | Loss meta-learning | | Benchmark [15] | |
|---|---|---|---|---|
| Slice | Hours | % of day (24h) | Hours | % of day (24h) |
| 1 | 7.28 | (30.34%) | 7.56 | (31.5%) |
| 2 | 24.00 | (100%) | 8.06 | (33.58%) |
| 3 | 24.00 | (100%) | 7.27 | (30.25%) |
| 4 | 19.41 | (80.88%) | 7.05 | (29.33%) |
| Average | 18.67 | (77.80%) | 7.485 | (31,18%) |

*2) Results:* The detailed implementation is described in Table **??**. We apply the automated loss meta-learning to the AC-RR task, and compare the results against a state-of-the-art solution for joint AC-RR in sliced networks [15]. The benchmark uses a Holt-winters algorithm to predict the future slice traffic, and then solves a disjoint optimization problem aiming at maximizing the revenue of the operator. We note that the expert-designed benchmark assumes knowledge of the relationship of revenue and decisions that may not be available in operational settings — whereas our approach does not.

We evaluate a scenario with four different SPs, each requesting a network slice with heterogeneous demanded capacities. The proposed loss meta-learning solution obtains a normalized profit of 0.1816, while the benchmark achieves 0.1678. Such non-negligible improvement of 8.22 percent is obtained in a setting where our solution *does not know* the relationship between profit and resource allocation decisions, while the benchmark does. Thus, we are able to improve over a carefully designed solution without any need for prior system knowledge and in a fully automated manner.

We also analyze the temporal dynamics of the AC-RR decisions. For that, we define the *average uninterrupted service time* (AVST) as the continued time interval during which a slice is served upon acceptance; the interval is thus concluded once the slice is torn down by the operator. The metric is averaged on a daily basis and expressed in hours, with a range from 0 (the slice is never allocated) to 24 (the slice is served all the time); it provides insights on the slice reliability, as SPs are interested in maximizing the uninterrupted time. Table **??** presents the AVST for each slice. Our model demonstrates substantial enhancements in AVST, positively impacting overall QoE, despite not being the direct focus of our model. Our solution is able to serve two of the slices continuously, as it solves a regression problem and has implicit temporal memory; instead, the benchmark incurs many interruptions as it solves an independent optimization problem at each time step.

The better performance of the loss meta-learning approach is the result of an extremely complex loss function that is learned from system observations. Figure 4 portrays three specific views of the complete eight-dimensional loss (with two dimensions per slice: the anticipatory decisions and the actual demand), and clearly illustrates how manually designing the loss would be impossible in this case.

## V. CONCLUSIONS

We propose one of the first solutions allowing for autonomous loss learning for generic regression problems. Our