Recursive meta-Reinforcement Learning for Personalized Sequential Dynamic Treatment Policies

BY

ELISA TARDINI
B.S., Politecnico di Milano, Milan, Italy, 2018

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Chicago, 2021

Chicago, Illinois

To Piera and Francesco

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Xinhua Zhang, for his continuous support during this last year. I would also like to thank Dr. G. Elisabeta Marai (UIC), Dr. David Fuller (University of Texas MD Anderson Cancer Center), Dr. Guadalupe Canahuate (University of Iowa), the Electronic Visualization Laboratory (UIC), and everybody else from UIC and the MD Anderson Cancer Center who contributed to this project for helping me in my first experience in academic research. I would also like to thank Dr. G. Elisabeta Marai and Dr. Pier Luca Lanzi (Politecnico di Milano) for serving on my committee.

I want to thank my family for (almost) always supporting me.

Finally, I want to thank all the friends I have made in the last year and a half in Chicago (special mention to the Palazzina), and all the other friends back in Italy and all over the world, whom I miss every day.

ET

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

A2C	Advantage Actor-Critic
AC	Actor-Critic
CC	Concurrent Chemotherapy
CECT	Contrast-Enhances Computed Tomography
DTR	Dynamic Treatment Regimes
DTS	Dynamic Treatment Simulator
GTVp	Gross primary Tumor Volume
HIPAA	Health Insurance Portability and Accountability Act
HNC	Head and Neck Cancer
IBEX	Imaging Biomarker EXplorer
IC	Induction Chemotherapy
LSTM	Long Short-Term Memory
MAB	Multi-Armed Bandit
MDACC	MD Anderson Cancer Center
MDP	Markov Decision Process
Meta-RL	Deep meta-Reinforcement Learning

LIST OF ABBREVIATIONS (continued)

ML	Machine Learning
OPC	Oropharyngeal Squamous cell Carcinoma
RL	Reinforcement Learning
RNN	Recurrent Neural Network
RSMRL	Recursive Sequential deep Meta Reinforcement Learning
RT	Radiotherapy
SL	Supervised Learning
SVC	Support Vector Classifier
VOI	Volume of Interest

SUMMARY

In recent years deep meta-reinforcement learning has extended the applicability of reinforcement learning (RL) algorithms: by integrating recurrent networks, trained models have the ability to quickly adapt to new unseen environments without the need for further backpropagation. These models, however, cannot adapt without having information on past rewards, and are therefore not directly applicable to a sequential decision-making setting in which multiple steps are required before observing the final reward.

One of the main applications affected by this limitation are dynamic treatment regimes, i.e. the problem of selecting the optimal medical treatment sequence for a patient at each step, keeping into account the complete past treatment history. By expanding deep meta-reinforcement learning to handle sequential decisions, a model would be able to prescribe the optimal treatment for each patient even if the patient's (or physician's) preferences on the outcome were never encountered by the model in training.

We propose a recursive deep meta-reinforcement learning approach which enables the model of each decision of the sequential process to learn from and adapt to unseen circumstances by recursively integrating the feedback of the models of other decisions in the process. We evaluate our approach on synthetic two-step processes with fixed transition probabilities but varying reward functions, to test the models' ability to propagate environment information from the final reward to intermediate steps. Finally, we train our model on a dataset of three-step chemo-radiotherapeutic and surgical treatment of oropharyngeal squamous cell carcinoma patients, proving our approach's ability to optimally handle previously unseen patient's preferences on survival and toxicity outcomes.

CHAPTER 1

INTRODUCTION

1.1 Meta-Reinforcement Learning

Deep meta-Reinforcement Learning (meta-RL) [1–3], is a novel addition to the RL field which, by adding Recursive Neural Networks (RNN) to existing RL algorithms, allows models to be applicable to a distribution of problems, rather than a single one, thus significantly expanding the environment adaptation capabilities of standard RL.

Meta-RL models have two fundamental components: the memory, and the standard RL model. The memory, in the form of an RNN, provides an embedding of the state space and past actions and rewards history, which constitutes the input to the standard RL model, which can be any traditional RL algorithm.

While traditional RL algorithms are trained in a single environment, and therefore can only act optimally in said environment, meta-RL models are trained on a distribution of environments with different characteristics (e.g. different reward functions or transition probabilities). It is the dynamic embedding of the environment provided by the RNN that allows the trained meta-model to assess its surroundings in real time and build a belief of which environment it is currently in, so as to act optimally with respect to the specific challenge.

State-of-the-art meta-RL algorithms, however, focus solely on Multi-Armed Bandits (MAB) environments or Markov Decisions Processes (MDP) with a single state space and an indefinite time horizon. These algorithms are therefore unfit to handle situations in which a fixed sequence of different actions must be performed before observing the final reward, as each junction in the sequence has a different state space representation and intermediate decisions have no immediate observed reward.

1.2 Dynamic Treatment of Oropharyngeal Squamous Cell Carcinomas

Dynamic Treatment Regimes (DTR) refer to "a sequence of decision rules, one per stage of intervention, that dictate how to individualize treatments to patients based on evolving treatment and covariate history" [4]. They are models (or sequence of models) whose goal is to determine the optimal treatment decision at each step for any patient, given the complete past medical history and a *goal* that defines what the desired outcome is. These models have grown in popularity in recent years as a *clinical decision support system* to aid physicians in assessing the correct treatment course of a variety of conditions, such as depression [5] and ADHD [6].

Head and neck cancer (HNC), which "includes cancers of the larynx, throat, lips, mouth, nose, and salivary glands" [7], is now an epidemic with 65,000 new cases in the US annually [8], whose treatment is, as in many other types of cancers, a dynamic and complex process. This therapy process involves making multiple, patient-specific treatment decisions, to maximize efficacy—e.g., reduction in tumor size, time of local region control, and survival time, while minimizing side effects [9–11].

For example, a specific patient may undergo radiotherapy alone (RT), radiotherapy with concurrent chemotherapy (CC), or induction chemotherapy (IC) [12]. After each round of IC, a decision must be made whether or not to continue IC or to start either radiotherapy

(RT) or concurrent chemotherapy (CC). These decisions are currently taken by clinician or multidisciplinary tumor boards based on pre-therapy patient characteristics or crude heuristics. Notably, current risk prediction models incorporated (e.g. AJCC Staging) in clinical decision support systems do not, by themselves, systematically direct clinicians to select an appropriate treatment that incorporates both oncologic and toxicity endpoints.

Further, disposition to initial induction chemotherapy is then followed by a second responsive disposition to either radiotherapy or concurrent chemoradiotherapy. Inferring the optimal treatment policies for multi-stage decisions (e.g., which treatment to give initially and then after observing treatment response, Figure 1) post hoc is challenging, as an optimal therapy sequence cannot be readily "pieced together" from several single-stage decisions.

For this reason, in the absence of rigorous clinical trials comparing adaptive induction chemotherapy permutations with concurrent radiotherapy, group comparison is exceedingly difficult, as simple models that account for confounders at initial disposition (e.g. propensity scores) are unequipped to incorporate sequential decision processes (e.g. the choice for chemotherapy concurrently after induction).

To address multi-stage models of therapy selection that incorporate both relevant cancer and side-effect considerations, and to allow personalization and optimization of health outcomes, we need to extend existing meta-RL algorithms to handle sequential decision making processes such as this dynamic treatment problem.

By leveraging a large number of oropharyngeal squamous cell carcinoma cases collected at a single institutional head and neck data tumor board at the MD Anderson Cancer Center

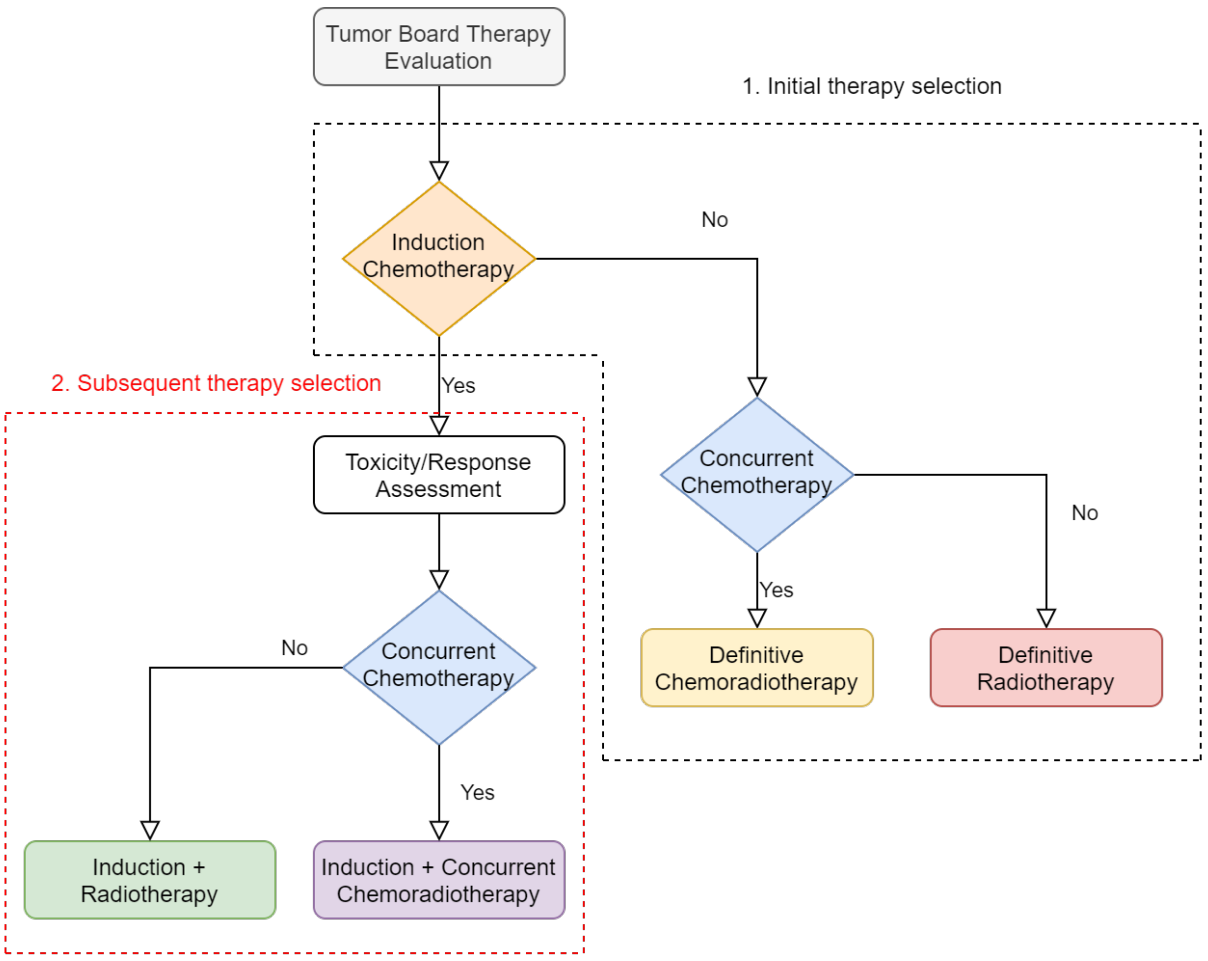

Tumor Board Therapy Evaluation
1. Initial therapy selection
Induction Chemotherapy
No
Yes
2. Subsequent therapy selection
Toxicity/Response Assessment
Concurrent Chemotherapy
No
Yes
Induction + Radiotherapy
Induction + Concurrent Chemoradiotherapy
Concurrent Chemotherapy
Yes
No
Definitive Chemoradiotherapy
Definitive Radiotherapy

Figure 1: Overview of the therapy selection process. The therapy selection process shows two distinct phases: initial therapeutic selection and subsequent therapeutic selection.

(MDACC), we propose an approach to leverage meta-RL as a method to construct a sequence of models as a clinical decision aid, by defining different reward functions as the patient's (or physician's) preferences over multiple clinically relevant outcomes from head and neck cancer patient-specific data. After training on multiple distributions of such weights, this sequential meta-RL model would be able to adapt to any patient-specific set of weights to prescribe custom optimal treatment decisions without the need for further customized training.

1.3 Recursive Sequential Deep meta-Reinforcement Learning

We designed a recursive meta-RL approach which extends existing meta-RL algorithms by recursively training a model for each decision of the sequential process with the incorporation of feedback from models of other decisions in the sequence. The generated sequence of models allows the feedback of the final reward to be propagated back to all models from last to first decision, while being appropriately filtered by consecutive models in the sequence.

Our approach was evaluated on a series of two-step synthetic MDPs with fixed transition and varying reward probabilities, which showed the ability of trained models to adapt to unknown environments without the need for additional training.

We then applied our novel approach to a dataset of oropharyngeal squamous cell carcinoma patients treated at MDACC as specified in Section 1.2. By modelling preferences on outcomes as different weights of a linear reward function, and training and testing our models on different distributions of weights, we obtained models capable of taking into account any set of preferences desired by the patient or physician, eliminating the need for further training of existing models or the development of custom-made models.

CHAPTER 2

RELATED WORK

2.1 Deep Meta-Reinforcement Learning

2.1.1 Introduction

Deep Meta-Reinforcement Learning (meta-RL) refers to methods in the Reinforcement Learning (RL) field which aim to expand the applicability and sample-efficiency of traditional RL algorithms, by training models that can rapidly adapt to a variety of seen and unseen tasks. This approach, that leverages the use of Recurrent Neural Networks (RNNs) as meta-learners added to existing algorithms, is not novel in the context of Machine Learning (ML) in general, as it has previously been applied to a variety of Supervised Learning (SL) problems [13], but has only recently been adapted to the RL setting [1, 2].

The goal of this meta-learning technique is to separate the two problems that any adaptive ML model must solve:

1. The *optimization problem*, which is the traditional goal of ML: for SL, this translates to minimizing the loss function between predicted output and ground truth, while for RL this translates to maximizing expected reward by finding the optimal policy for each state. This problem is solved by the traditional ML algorithm part of the meta-learning architecture.

2. The *adaptation problem*, which is the goal of the meta-learner: by keeping a memory of past inputs and outputs of the model, and their resulting loss/reward, the RNN constructs a belief on which of the tasks in a given family is currently being solved, and can therefore output a representation biased by such belief. This biased representation is then fed to the traditional learner for outcome optimization.

A model that solves these two problems can therefore be optimal in a distribution of tasks, rather than a single one, since it can adapt its belief on the task currently at hand and act accordingly. This is useful for two reasons:

- *Applicability*: because the model can adapt to multiple tasks, regardless of whether these tasks were encountered during training or not, we no longer need a different model for each task, but just one meta-model for all.
- *Sample efficiency*: because the model can quickly adapt to previously unseen tasks by exploiting the knowledge it has on similar tasks, we do not need to retrain a model from scratch every time we encounter a new setting, which would require a significant amount of observed samples.

These meta-learners are therefore closer than traditional ML algorithms to the learning process of humans, as humans, when facing a new problem, incorporate the knowledge they already have about similar problems to quickly learn the new task (e.g. incorporating the knowledge about how to walk when learning how to run).

2.1.2 Markov Decision Processes

The goal of a traditional RL algorithm is to "solve" a Markov Decision Process $MDP = \langle \mathcal{S}, \mathcal{A}, P, R, \gamma, \mu^0 \rangle$, where:

- $\mathcal{S}$ is a set of *states*.
- $\mathcal{A}$ is a set of *actions*.
- $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}_+$ is the *transition probability distribution*, where $P(s'|a, s)$ is the probability of transitioning to state s' from state s when performing action a.
- $R : \mathcal{S} \times \mathcal{A} \rightarrow [-R_{max}, R_{max}]$ is the *reward function*, which maps each state-action combination to the expected reward, $R(s, a) = E[r|s, a]$.
- $\gamma \in [0, 1]$ is the *discount factor*, which determines the relevance of past/future rewards based on their distance in time.
- μ^0 is the *initial state distribution*, where $\mu_i^0 = P(S_0 = i)$ for any state i

"Solving" an MDP means finding the optimal policy $\pi^* : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}_+$ that maximizes expected return.

The *state-value function* associated to a policy π, $V^\pi : \mathcal{S} \rightarrow \mathbb{R}$, is the expected return starting from any state $s \in \mathcal{S}$ and following policy π. The *action-value function* associated to a policy π, $Q^\pi : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, is the expected return starting from any state $s \in \mathcal{S}$, taking action $a \in \mathcal{A}$ and then following policy π [14].

2.1.3 Architecture

Meta-RL algorithms exploit their meta-learning capabilities to solve a distribution $\mathcal{D}$ of MDPs, rather than a single one. The MDPs in the distribution might differ in:

- *Only reward* probabilities $R \in \mathcal{R}$ (keeping transition probabilities fixed)
- *Only transition* probabilities $P \in \mathcal{P}$ (keeping reward probabilities fixed)
- *Both reward and transition* probabilities

The architecture of a meta-RL model is presented in Figure 2.

As mentioned in Section 2.1.1, the traditional RL model is stacked on top of one (or more) RNN, forming the meta-model. In Figure 2 the represented RL model is an actor-critic (AC) model, which outputs both the policy $\pi_t(x_t)$ and the state value $V_t(x_t)$ for the current state x_t, but it can actually be any traditional RL model. The inputs to the RNN (and therefore to the meta-model) at timestep t are:

- Current *state representation* x_t: this gives the model knowledge of its current surroundings. The state may also undergo different encodings before being fed to the meta-model (e.g. through some convolutional layers in the case of images, not pictured in Figure 2).
- Current *timestep* t: this gives the model knowledge of how far along in the episode it currently is.
- *Previous action* a_{t-1}: the action performed by the model at the previous timestep.
- *Previous reward* r_{t-1}: the reward resulted from performing action a_{t-1} in the current environment.

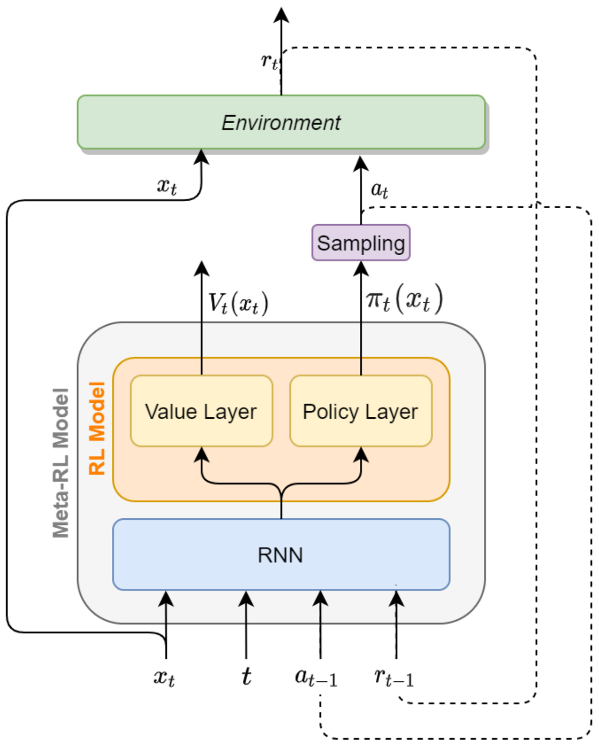


Figure 2: Architecture of a meta-RL model

These inputs are processed by the RNN and the resulting embedding is passed to the traditional RL model, which outputs the chosen action a_t for the current timestep.

The meta-learning capabilities of the model are given by incorporating past action-reward combinations (a_{t-1}, r_{t-1}) in the input of the RNN: by keeping track of these values, the RNN can build a belief on the environment it is currently in and pass this belief to the RL model, which can then choose the optimal policy accordingly.

Notice that there is no need to include the previous state x_{t-1} in the input as it has already been part of the input at timestep $t-1$ and has therefore already influenced the hidden state of the RNN.

2.1.4 Training

The training phase of meta-RL is divided in episodes of fixed length: at the beginning of each episode, an MDP is sampled from distribution $\mathcal{D}$ and held constant until the end of the episode; each of the trials in an episode constitutes a single run of the model in the current environment. As the model interacts with the environment throughout the episode, it builds a stronger belief on the sampled MDP, and its performance improves. At the end of the episode backpropagation is performed, and the cycle starts again in a new episode. Since it's the hidden state of the RNN that represents the belief on the current environment, it needs to be reset (i.e. set to 0) between each episode, as in each episode the sampled environment is different.

once training is completed, the model is then evaluated again on episodes of fixed length, with a different environment being sampled at the beginning and the state being reset at the end, but now there is no backpropagation, the weights are fixed.

2.1.5 Results

Models trained and evaluated on Multi-Armed Bandits (MAB, i.e. MDPs with a single state and n actions with associated reward probabilities) showed a sub-linear cumulative regret curve across evaluation episodes (Figure 3), suggesting that they can in fact build a belief on the environment and act optimally accordingly. They also performed better than some theoretically optimal MAB algorithms, such as Thompson Sampling and UCB [1, 2] (see Figure 3). Both

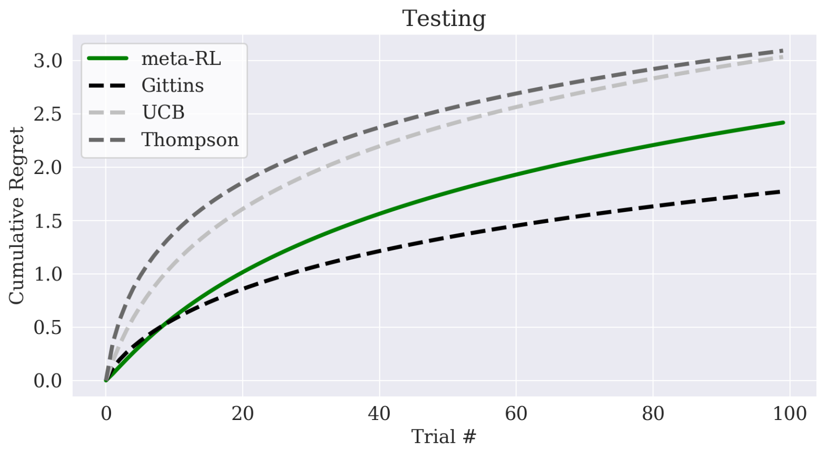


Figure 3: Average cumulative regret per trial of a meta-RL model, as compared to other optimal algorithms

papers highlighted the relevance of including previous reward r_{t-1} in the input: without it, performance was at chance level, which meant that the model wasn't able to properly build a belief on the environment and exploit its bias.

When it comes to evaluation of MDP models, Duan et al. [1] focused on tabular MDPs, once again comparing the meta-model's performance to known algorithms. Wang et al. [2] had a different approach, as they focused on the "two-step task", an experiment derived from the neuroscience field [15]: in this task (in the version presented by Wang et al., presented in Figure 4), the choice between two actions leads to either of two second-stage states, to which is associated a certain reward probability. The transition probabilities are fixed throughout training and testing, but the reward probabilities are randomly assigned at the beginning of each episode: both second-stage states have the same reward value, 1, but one is assigned probability of reward 0.9, the other 0.1.

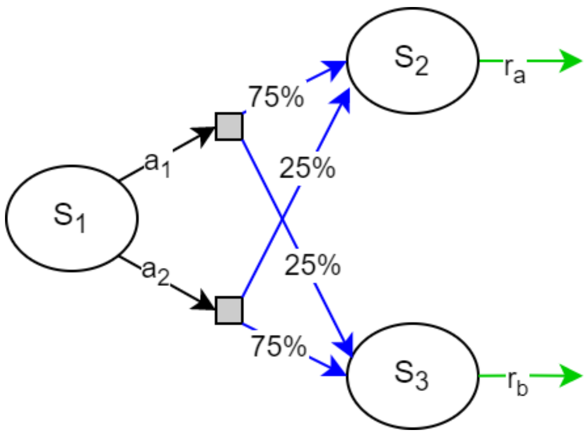


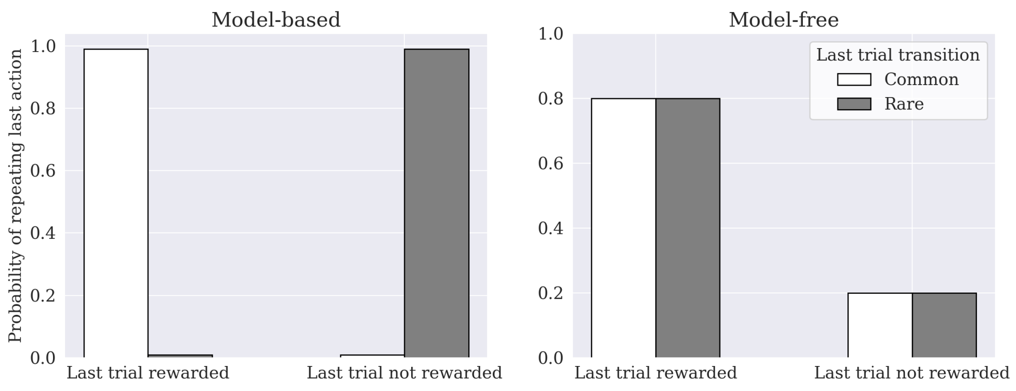
Figure 4: The two-step task: starting from state S_1, actions a_1 and a_2 lead to second-stage states S_2 and S_3 with different transition probabilities (pictured in blue). States S_2 and S_3's reward probabilities, r_a and r_b, are randomly assigned at the beginning of each episode

The goal of this task is to differentiate between a model-based and a model-free behaviour:

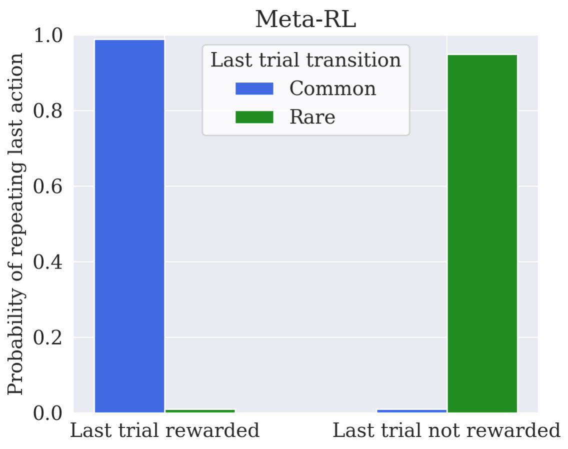
- A *model-based* behaviour takes into account transition probabilities when assessing the result of an action: intuitively, if a common transition ($p = 0.75$) led to reward, the optimal policy would be to repeat the previous action; on the other hand, if a rare transition ($p = 0.25$) led to reward, the optimal policy would be to change the action, as the other action would have a higher probability of leading to the previous state. In other words, a model-based behaviour is able to discern the contribution of state transitions from that of reward probabilities to the final reward.

- A *model-free* behaviour does not take into account transition probabilities when assessing the outcome of an action: an action that led to reward is more likely to be repeated than one that didn't, regardless of the resulting state.

The results of Wang et al. (Figure 5) show that, interestingly enough, a meta-RL model, which is trained based on a model-free RL algorithm, still exhibits model-based behaviour.



(a) Model-based vs. model-free behaviour: in a model-based behaviour the probability of repeating the last action changes based on the last reward and the probability of the last transition, while in a model-based behaviour the probability only depends on the last reward.



(b) Behaviour of a meta-RL model

Figure 5: Model-based vs. model-free behaviour in the two-step task: a meta-RL model trained with a model-free RL algorithm still exhibits model-based behaviour

2.2 Dynamic Treatment Regimes with Multiple Reward Functions

2.2.1 Dynamic Treatment Regimes

Dynamic Treatment Regimes (DTR) are models trained to provide optimal personalized treatment for a clinical patient in a (possibly sequential) therapeutic course: at each treatment junction, the model must prescribe the most suitable treatment given the patient's history in order to provide them with the best possible outcome. In the case of a multi-step sequence, the patient's history at each junction also includes past treatment decisions and their intermediate results.

The most suitable ML models to be applied to this category of clinical problems are RL models, as we can encode the therapeutic course as an MDP, where:

- The *current state* is the patient's history at junction i, H_i.
- The *actions* are the treatment decisions.
- *Transition probabilities* map treatment decisions and patient's history to intermediate or final outcomes of the treatment sequence.
- *Reward probabilities* are the probability of a positive outcome given patient's history and prescribed therapy.

The optimal treatment sequence can then be found by training any RL algorithm on the MDP defined above.

However, the fact that the therapeutic course is sequential and the state at each junction contains the complete history means that each treatment decision has a separate state representation, which in turn means that we cannot model the complete sequence as one MDP, but rather we have a separate MDP for each junction, consisting of exactly one step.

As a result, we need to train a different RL model for each decision, which poses a problem for existing RL algorithms, as only the final treatment junction has a proper reward associated (e.g. survival), while all the previous decisions only have intermediate results, whose connection to the final outcome is usually not straightforward.

The way this limitation has been overcome [4–6, 16] is by training the models for each treatment junction recursively, starting from the last one: the last one can be trained with any traditional algorithm without a need for adjustments, as the final outcome (reward) is a direct consequence of the last decision; the second-last decision's model is then trained with a "fake" reward which corresponds to the value of the state to which the decision leads, as computed by the (already trained) last decision's model:

1. The last decision d_N's model is trained with a standard RL algorithm, using the final outcome of the treatment sequence.
2. The second-last decision d_{N-1} is trained again with a RL algorithm, but instead of the final outcome the metric used as reward is the value of the next state $V(s_N)$ to which decision d_{N-1} led, as computed by the model of decision d_N.
3. The training process is recursively repeated, with the third-last decision d_{N-2}'s model being trained on the value $V(s_{N-1})$ computed by decision d_{N-1}'s model and so on, until the initial treatment decision.

Intuitively, the goodness of an intermediate decision is equal to the goodness of the state to which is leads, which, being a starting state for the following decision, can be computed by the value function of the following decision's model.

The training process of a generic 3-step DTR is shown in Figure 6.

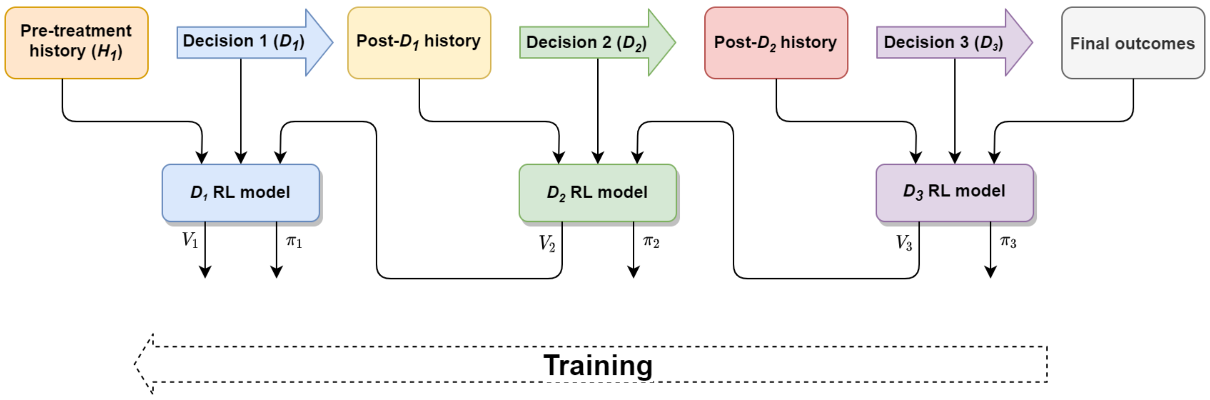


Figure 6: Training of a 3-step Dynamic Treatment Regime: the third decision model (D_3) is trained on the final outcome, then the second decision model D_2 is trained on the value V_3 computed by the D_3 model, and the first decision model D_1 is trained on the value V_2 computed by the D_2 model.

Because this approach requires the RL models to output the value of a given state, existing literature [4–6,16] mostly focused on Q-learning, since by learning the Q-function we can directly compute the value function and the optimal action. These authors focused on a version of Q-learning different from the "traditional" one, as it learns directly from observational data by modelling the Q-function as a regression problem, without interacting with the environment:

this is the main limitation of applying RL to the medical field, as it is not possible to directly apply decisions made by untrained models to real patients.

One limitation of the current approach on learning DTR is that the desired outcome to be optimized must be statically defined before training, and cannot be dynamically adjusted to a patient's (or physician's) preferences. For example, an oncological therapy course needs to balance two main outcomes:

- *Survival rate*: in order to be as effective as possible in terms of survival, a treatment must usually be very aggressive when it comes to chemo-radiotherapeutic as well as surgical options.
- *Quality of life*: a very aggressive treatment, on the other hand, tends to have more serious side effects, such as chemotherapeutic toxicities and surgical complications, which might seriously impact the patient's quality of life during and after treatment.

These two outcomes are obviously in contrast with each other, as maximizing one tends to limit the other. For this reason, different patients might have different preferences on their relative importance, which would lead to different treatment decisions.

If we followed the Q-learning approach, in order to have the optimal DTR for any set of preferences we would need to train a different model for each possible set of preferences, which is clearly infeasible even if we limit ourselves to a linear combination of outcomes.

2.2.2 Learning with Multiple Reward Functions

Some new approaches have been proposed to try to model multiple reward functions in DTR settings, in order to be able to account for varying preferences of patients on outcomes.

Barret et al. [17] propose a variation of the value iteration algorithm that can model the optimal treatment for any linear combination of preferences over N outcomes, expressed as a vector $\vec{w} = [w_0, w_1, ..., w_N]$. The outcomes are also represented as a vector $\vec{r} = [r_0, r_1, ..., r_N]$. We can then obtain a scalar reward by performing the dot product of preferences and outcomes:

$$r = \vec{w} \cdot \vec{r} = w_0 * r_0 + w_1 * r_1 + ... + w_N * r_N \tag{2.1}$$

Their findings showed that any Q-value that is optimal for some $\vec{w}$ lies on the convex hull of all possible Q-values: from this observation they built their modified version of the value iteration algorithm.

This work has multiple limitations: firstly, it only limits preferences on outcomes to linear combinations, and secondly, their algorithm requires finite and small state spaces, as well as true (or estimated) transition and reward probabilities.

Lizotte et al. [18, 19] tackled some of these limitations by further generalizing the value iteration algorithm to handle continuous state spaces. The preferences are once again a linear combination of outcomes, and the resulting algorithm relies on the concepts of convex hull of Q-values and upper convex envelope of Q-functions.

Although this approach resolves the limitation to small finite state spaces, it still requires known (or estimated) transition and reward probabilities and can only model linear preferences.

CHAPTER 3

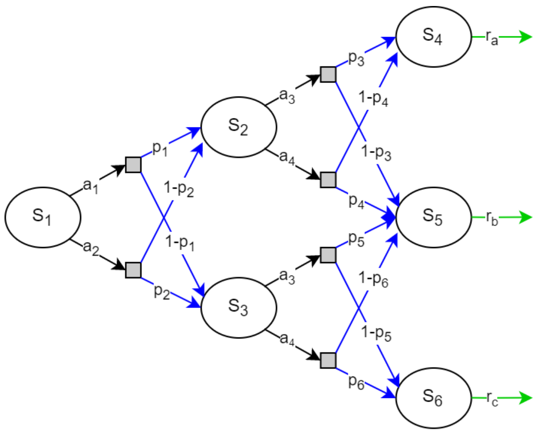
MOTIVATION

3.1 The Three-Step Problem

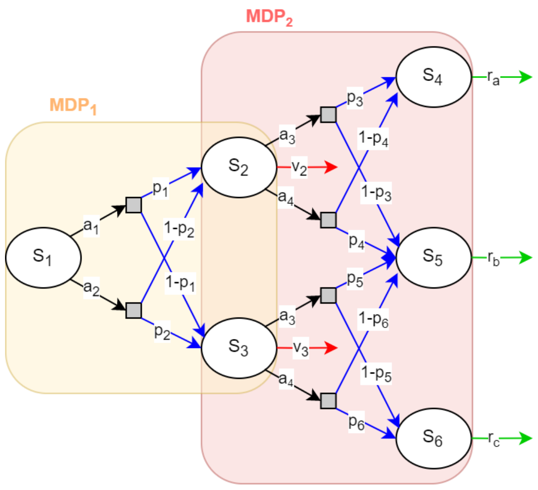
To introduce our novel idea and give a better understanding as to why existing meta-RL algorithms cannot be used in this setting, we introduce as a toy example the “three-step task”, an extension of the “two-step task” presented in Section 2.1.5.

In this environment, represented in Figure 7a, we have added an extra step (i.e. an extra action) to the two-step task: the first action's MDP structure (MDP_1) is exactly the same as in the two-step task, but second-stage states S_2 and S_3 are no longer final states, but rather the initial states of the second-step MDP (MDP_2), with actions a_3 and a_4 leading to third-stage states S_4, S_5, and S_6 with probabilities p_3, p_4, p_5, p_6, $1 - p_3$ etc., depending on whether the starting state is S_2 or S_3. Reward probabilities r_a, r_b, and r_c are then associated to the third-stage states.

Notice that this MDP is an extension of the two-step task not just because of the extra step, but also because we generalized transition probabilities p_1, p_2, p_3, p_4, p_5, and p_6, as well as reward probabilities r_a, r_b, and r_c. This way, we can use this MDP structure to train models for each of the 3 categories of MDP distributions presented in Section 2.1.3, by varying either r (1st category), p (2nd category), or both (3rd category).



(a) Complete MDP structure of the three-step task



(b) The two two-step MDPs that constitute the three-step task

Figure 7: MDP structure of the three-step task, as a complete process (a) and as two separate two-step tasks (b)

Given the key role that past rewards have in the working of existing meta-RL algorithms [1, 2], we cannot directly apply these algorithms to this kind of MDP, since we only get a reward after the second-stage actions (a_3 and a_4), so the first-stage actions (a_1 and a_2) have no immediate reward associated to them.

A first approximation could be to train two separate models, one for first-stage actions and one for second-stage actions, and use the final reward in both. This, however, would not be accurate for the first-stage, since the final reward doesn't depend on the first-stage action alone, but also on the second-stage action. This would mean that an optimally trained first-stage model, if followed by a badly trained second-stage model, would not be able to act optimally, because the second-stage model would lead to bad outcomes regardless of whether the first-stage model chooses the optimal action or not, so it would be harder for it to discern between optimal and non-optimal actions.

The goal is to find a feedback to provide to the first-stage model that accurately assesses the optimality of the model's actions, in the same way the reward does for a final-stage decision, and that can therefore serve the same purpose as the (unavailable) final reward.

3.2 The Dynamic Treatment with Multiple Rewards Problem

As specified in 2.2, standard RL algorithms cannot train models capable of adapting their decision-making process to custom patient preferences when applied to Dynamic Treatment Regimes (DTR): some extensions have been made [17–19], but they are still very limited, as they require finite small state spaces and linear preferences.

Since the goal of meta-RL is to be able to act optimally in a distribution of environments, by modelling different preferences over clinical outcomes as different reward functions and training a meta-RL model on this distribution of preferences we would have a single meta-model capable of handling any patient-specific preference, linear or non linear, already known or previously unseen. This would therefore remove all the limitations of the current literature on multiple reward functions, and provide a general adaptive model which would be a closer approximation of the decision-making process made by human physicians.

However, as shown in Section 3.1 with the three-step task, state-of-the-art meta-RL cannot be applied to sequential decision-making processes such as those arising in the DTR setting. There is therefore a need to find a new approach to meta-RL capable of expanding algorithms to finite fixed interdependent sequences of decisions.

3.3 Goals

Our goal is to expand the applicability of state-of-the-art meta-RL algorithms by adapting them to a fixed-sequence decision-making setting, such as that of clinical therapeutic courses.

We propose to incorporate into existing meta-RL models a recursive approach similar to that currently used in DTRs, as it has been proven valid in the standard RL setting to handle such sequential dependencies.

Our ultimate aim is to apply such novel recursive meta-RL approaches to DTRs to model patient-specific preferences over treatment outcomes as a distribution of environments. This application would lead to a generalized model capable of adapting to any set of outcome preferences, regardless of whether they were previously encountered in training or are completely new.

CHAPTER 4

DATASET

4.1 Oropharyngeal Squamous Cell Carcinoma Dataset

A curated HNC dataset of oropharyngeal squamous cell carcinoma (OPC) patients treated at MD Anderson Cancer Center between 2005 and 2013 was analyzed in this project. All methods for this study were performed in accordance with the University of Texas MD Anderson Cancer Center IRB guidelines and regulations. The UIC IRB determination can be found in Appendix A. Being a Health Insurance Portability and Accountability Act (HIPAA)-compliant retrospective study, the prerequisite for informed consent was waived. The complete treatment process is summarized in Figure 8.

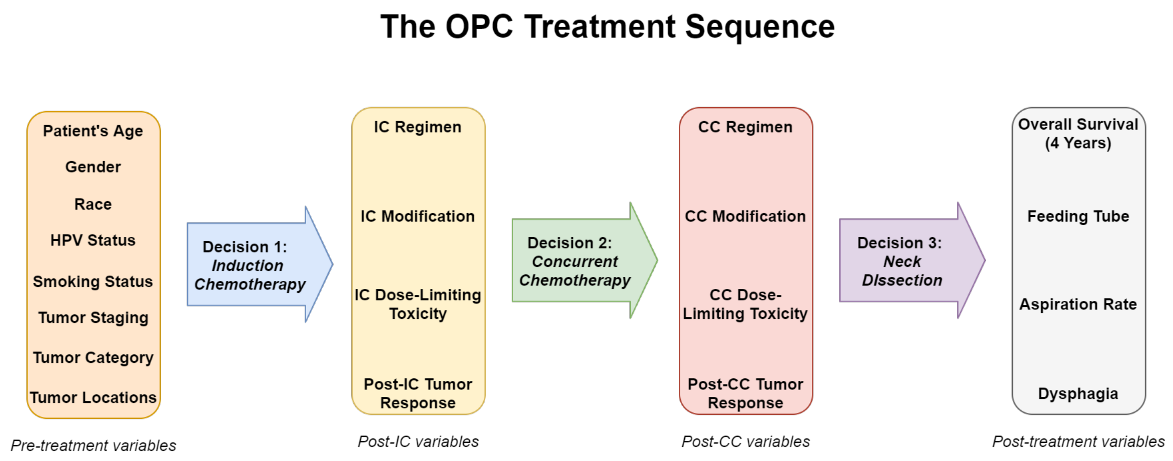


Figure 8: Main features, outcomes, and decisions of the OPC treatment sequence

Clinical features recorded at diagnosis including age at diagnosis, sex, ethnicity, HPV status, smoking status and frequency, subsite of the primary tumor within the oropharynx, T category, N category, therapeutic combination, AJCC stage (8th edition), as well as intermediate results of the treatment decisions, such as Dose-Limiting-Toxicity (DLT) and tumor response, measured between treatment decisions were extracted from electronic medical records. Tumor response is included as a categorical variable (complete/partial/stable disease for primary tumor and nodal involvement). Table V in Appendix B shows the demographics of patients for the main clinical features and outcomes considered. Attributes with missing data are also identified in the table.

Furthermore, the dataset also included radiomic features extracted from contrast-enhanced computed tomography (CECT) at initial diagnosis, prior to any active local or systemic treatment. The details on feature extraction and preprocessing are presented in Appendix , as cited from Tosado et al. [20] These radiomics features were not included in the RL models because, despite the abundance of research on radiomics models for head and neck cancer [21–29], preliminary analyses (such as the Dynamic Treatment Simulator described in Section 5.2 and the training and evaluation of traditional RL models) showed that such features did not improve performance, and instead mostly worsened it and increased variance.

For each patient, post-treatment outcomes include survival and overall toxicity and toxicity outcomes such as feeding tube and aspiration rate. As a survival outcome we consider Overall Survival at 4 years. No blind assessment of any of the considered outcomes was performed, therefore only patients with enough follow-up time (at least 4 years) or who died before 4 years were considered for the analysis. We focus our analysis on two outcomes: Overall Survival at 4 years and Dysphagia, which refers to the presence of either feeding tube or aspiration 6 months after treatment.

For the purpose of this project, we furthermore consider three treatment decision points for each patient as part of the treatment policy: Decision 1 (D1): Induction Chemotherapy (IC) or Not; Decision 2 (D2): Concurrent Chemotherapy (CC) or Radiotherapy alone (RT); and Decision 3 (D3): Neck Dissection (ND). These are the treatment decisions that we want our RL models to optimize.

4.2 Preprocessing

The 536 samples of the dataset were split into two distinct sets for training and testing using a 75-25% random split.

No blind assessment of the decisions or outcomes was made. Unknown HPV status was handled using a distinguished value (0). Missing values for all other covariates were handled using single imputation: median for numerical variables, mode for categorical ones.

The ordinal covariates pathological grade, T-category, N-category, AJCC, prescribed chemo (none/single/doublet/triplet/quadruplet) were coded as numerical features.

After these preprocessing steps, all features were rescaled in the [-1,+1] range, as is standard procedure when training neural networks.

CHAPTER 5

METHODS

5.1 Recursive Sequential Deep Meta-Reinforcement Learning

Our plan is to use the same recursive approach described in Section 2.2, which has been shown to work in traditional DTR settings, to extend meta-RL algorithms to sequential problems.

Going back to the three-step task example (Figure 7), this means decomposing the sequential MDP in two two-step tasks, one for the first-stage and one for the second-stage actions:

- The second-stage two-step task MDP_2 is a standard two-step task, as it has immediate reward associated to action and transition, and its model can therefore be trained using the standard existing meta-RL algorithms.
- The first-stage two-step task MDP_1 needs to associate surrogate rewards to states S_2 and S_3 (v_2 and v_3 in Figure 7b, respectively), which are computed by the MDP_2 model as the values of states S_2 and S_3, respectively. Once this computation has been performed, model training can be carried out with the standard existing meta-RL algorithms.

In other words, Recursive Sequential Deep Meta Reinforcement Learning (RSMRL) divides the training of a sequential MDP model into basic sub-models, one for each decision junction, which are recursively trained backwards through standard meta-RL algorithms using as last reward the value of the state to which the last action led, as computed by the following decision junction's model.

The resulting meta-model architecture for a two-decision MDP such as the three-step task presented in 3.1 is presented in Figure 9.

The basic models, unchanged from the state-of-the-art meta-RL models (one or more RNNs followed by a standard RL algorithm), are now stacked to account for all the decisions in the sequence: between the first and the second model, however, there is the need for interaction with the environment, which, given the action outputted by the first model, will give the resulting state, i.e. the starting state of the second model.

The novelty of this approach is not just in the stacking of meta-RL models as building blocks of a more complex model, but is also in the new input to the single cells, which incorporates more feedback from the environment as well as from other models in the stack. In particular, the new types of feedback are:

- *State feedback* x_{t-1}^{j+1}: for each model j in the stack, the previous *resulting state* of the action chosen by model to be performed in environment MDP_j. This incorporates more feedback from the environment.
- *Value feedback* $V_{t-1}^{j+1}(x_{t-1}^{j+1})$, $(j < N)$: for each model j except the last one (N) in the sequence, the reward r_{t-1} is substituted by the value computed by the following model in the sequence. This creates a chain of feedback between models in the sequence, from last to first.

Notice that state-of-the-art meta-RL models did not include state feedback as part of the input, as in single MDPs the resulting state of the past action is simply the current state x_t, so it would have been redundant information.

In RSMRL, instead, the current state and the previous resulting state no longer coincide, as they belong to two different sequences and, more importantly, two different and disjoint state spaces (i.e. the starting states of two different MDPs). This new feedback from environment to model is thus necessary for the model to properly assess its surroundings and learn from it, and is fundamental in order for it to exhibit the model-based behaviour of which meta-RL models are capable.

The incorporation of value feedback is, on the other hand, the truly novel part of the RSMRL approach, as it adds a new kind of feedback to the environment-to-model one: the *model-to-model feedback*. This recursion is what truly makes the sequential decision making possible, as it allows the chained models to propagate information about the final reward to all previous decisions back to the first, while being appropriately filtered to represent the "partial" merits of each model in achieving the final reward. Using the state-value function instead of the reward leads to a very cohesive sequence of models, that learn and adapt together, with a reduced chance of inconsistencies or divergences.

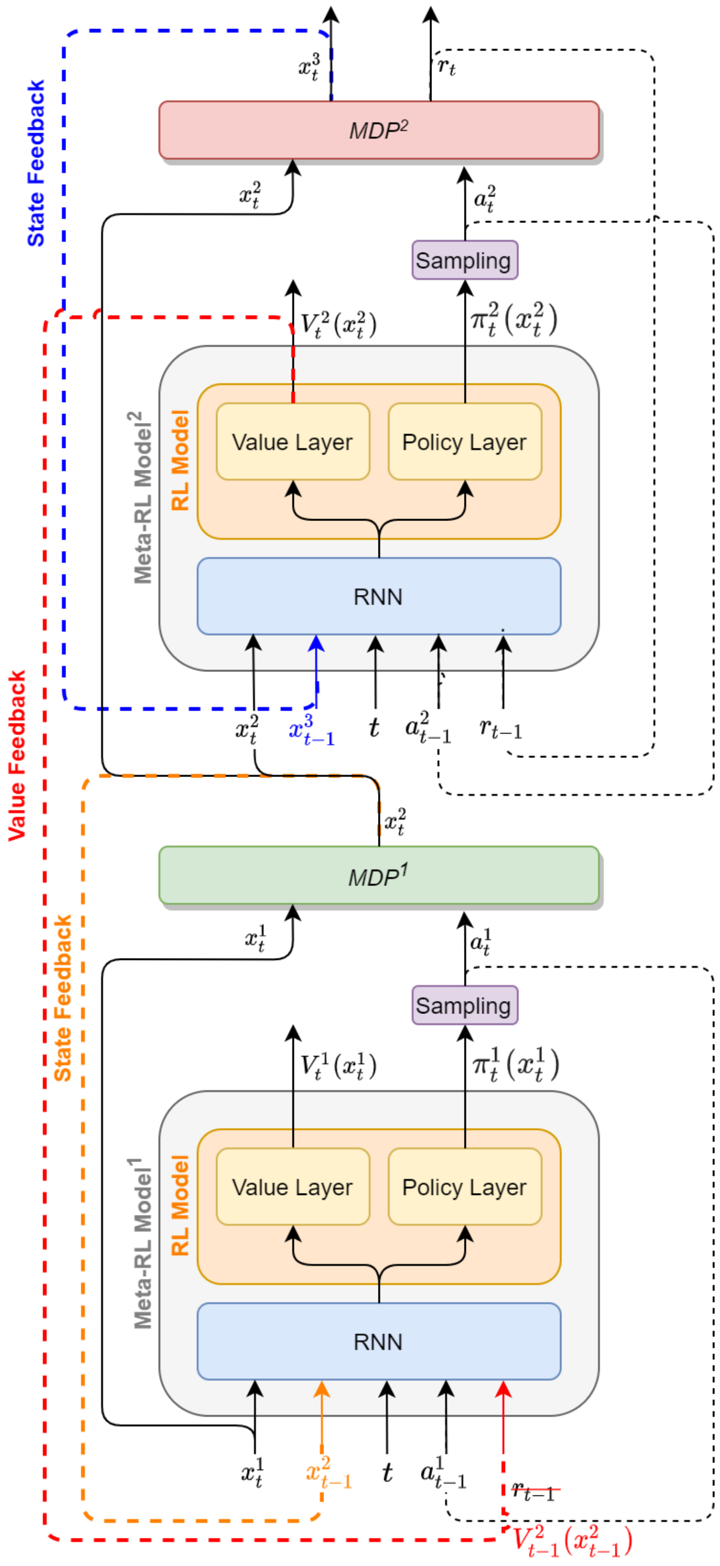


Figure 9: Architecture of a recursive sequential meta-RL model for a two-decision MDP. We have two new types of feedback: from environment to model (x_{t-1}^2, in orange, x_{t-1}^3, in blue), and from following value to previous model ($V^2t-1(x_{t-1}^2)$, in red).

5.1.1 Training

When it comes to choosing how to train the sequence of RSMRL models, there are two options:

- Training *all the models together*: each training trial is a complete sequence from starting state to final reward, using the actions chosen by all the models. At the end of the episode (i.e. series of trials in the same environment), backpropagation is performed on all the models.
- Training *one model at a time*, from the last to the first one: this is the approach followed by the literature on DTR when training on observational data [4–6, 16], in which the last model of the sequence is trained first on the final reward, then the second-last model is trained on the value computed by the last (already trained) model, and so on.

Both these options have some positive and some negative implications.

Training all the models together seems like the more natural choice, as the sequence of models would be learning at the same time and through exactly the same episodes, giving more coherence to the model as a whole.

However, there is a concern about error propagation: at the beginning of training, the value feedback from model to model is given by untrained models, so it is very likely an inaccurate representation of the state value. This could cause models that are earlier in the sequence to not be able to learn properly because of the amplification of error through the recursive chain.

Another possibility is that the error propagation does not prevent correct learning, but just slows it down, as models could possibly only start to learn properly once the values provided by their following model become accurate (i.e. once the subsequent model's training has reached convergence). In this case, the final process would be equivalent to training one model at a time, but with a greater expense of time and resources, as at each episode we need to use and backpropagate all models but only one is actually learning.

Training one model at a time would, on the other hand, removes this error propagation problem, as the only values propagated would be computed by already trained models, and would therefore be as accurate as possible, without the overhead of training all the models together.

But this alternative does not come without limitations. In particular, training one model at a time requires a model selection step between each model's training: it is not guaranteed that the final model is the optimal one, as it might have overfitted, so we would need to choose a performance metric to evaluate at each step during training, periodically save intermediate models during training, and once training is completed choose the best-performing one, either manually or automatically.

With these considerations in mind, for our approach we chose to *train all models at the same time*, so as to avoid the problems of choosing an accurate performance measure for in-training model selection and to maximize the applicability of our approach, without restricting it to problems for which it is possible to start an execution from an intermediate state.

The final algorithm of our approach is presented in Algorithm 1. As in standard meta-RL, training is divided in episodes, each episode being divided in a fixed number of trials, i.e. runs of the algorithm. The environment is sampled at the beginning of each episode and held constant throughout: this way, the model has the duration of the episode to build a belief on the environment it currently is in. Backpropagation is performed at the end of each episode.

As already mentioned in Section 3.1, environments in a distribution must share the state space and actions representations, but can differ in transition and/or reward probabilities.

Algorithm 1: Recursive Sequential Deep Meta-Reinforcement Learning (RSMRL)

```
Data: Distribution $\mathcal{D}$ of environments $MDP = [MDP^1, MDP^2, ..., MDP^N]$
      composed by $N$ one-step MDPs
Number of training episodes $N_E$
Number of trials per training episode $N_T$
Result: Trained RSMRL model $M = [M_1, M_2, ..., M_N]$ composed by $N$ one-step
        models
for $e \leftarrow 1$ to $N_E$ do
    /* Reset the hidden state of all the models                              */
    for $i \leftarrow 1$ to $N$ do
        $ResetState(M_i)$
    end
    /* Initialize values of past action, reward, and resulting state         */
    for $i \leftarrow 1$ to $N$ do
        $a_0^i \leftarrow 0$
        $r_0^i \leftarrow 0$
        $x_0^{i+1} \leftarrow 0$
    end
    /* Sample new environment                                                */
    $MDP_e \leftarrow Sample(\mathcal{D})$
    /* Explore environment                                                   */
    for $t \leftarrow 1$ to $N_T$ do
        Pick random initial state $x_t^1$
        for $i \leftarrow 1$ to $N$ do
            $\pi_t^i, V_t^i \leftarrow M_i(x_t^i, x_{t-1}^{i+1}, t, a_{t-1}^i, r_{t-1}^i)$
            $a_t^i \leftarrow Sample(\pi_t^i)$
            if $i < N$ then
                $x_t^{i+1} \leftarrow MDP_e^i(x_t^i, a_t^i)$
            else
                $x_t^{i+1}, r_t \leftarrow MDP_e^i(x_t^i, a_t^i)$
            end
        end
        $r_t^N \leftarrow r_t$
        /* Compute value feedbacks                                           */
        for $i \leftarrow N-1$ to $1$ do
            $r_t^i \leftarrow V_t^{i+1}(x_t^{i+1})$
        end
    end
    /* Backpropagate each model at the end of the episode                    */
    for $i \leftarrow 1$ to $N$ do
        $loss_e^i \leftarrow ComputeLoss(\vec{r^i}, \vec{a^i}, V^i(\vec{\vec{x}}), \vec{\pi^i})$
        $Backpropagate(M_i, loss_e^i)$
    end
end
```

5.2 Dynamic Treatment Simulator

Because we need an environment to train and evaluate our recursive meta-RL models when applied to DTR problems, our evaluation includes building a separate model which, given a patient's history and the prescribed treatment, predicts the outcome of that treatment. This patient treatment *digital twin* approach enables us to simulate the results of applying our models to patients.

To this end, we built a Dynamic Treatment Simulator (DTS), which, based on the patient's history and treatment decisions, predicts the outcome at the next step. In order to give a larger relevance to the treatment decision in determining the next-step outcome Y_{i+1}, the input to the DTS is not simply the patient's history H_i and the treatment decision A_i (which has either value $+1$ or -1), instead it is the concatenation of patient's history and patient's history multiplied by the treatment decision value:

$$Y_{i+1} = DTS(H_i, H_i * A_i) \tag{5.1}$$

This construct was inspired by the literature on building RL models from observational data to model Dynamic Treatment Regimes [4–6,16], as described in Section 2.2: by including $H_i * A_i$ instead of A_i as input, half of the input variables are influenced by the decision value, instead of just one. This prevents the prediction model from predicting the same outcome regardless of the decision value, as this would lead to a pointless environment to train a decision-making algorithm.

The simulator contains a separate model for each intermediate and final outcome measure, built using a Support Vector Classifier (SVC) and tuned via 5-fold cross-validation over the training data to select the best kernel (between gaussian, sigmoid, or polynomial) and hyperparameters. The output is a probability distribution over possible outcomes computed using Platt scaling [30]. These models serve as an *in silico* digital twin of the patient treatment, as we can use them to dynamically simulate the patient's *in vivo* course as a function given treatment policy, without physically having to treat the patient. The full details of the SVC are provided in Table I.

TABLE I: HYPERPARAMETER DETAILS OF ALL MODELS IN THE DYNAMIC TREATMENT SIMULATOR.

Outcome	*C*	*Kernel*	*Degree*	*Gamma*	*Class Weight*
Prescribed Chemo	3	polynomial	5	automatic	balanced
Chemo Modification (Y/N)	1	gaussian	-	automatic	balanced
Dose modified	4	polynomial	14	automatic	balanced
Dose delayed	60	gaussian	-	automatic	balanced
Dose cancelled	11	gaussian	-	automatic	balanced
Regimen modification	3	polynomial	20	automatic	balanced
DLT (Y/N)	2	gaussian	-	automatic	balanced
DLT_Dermatological	2	polynomial	15	automatic	balanced
DLT_Neurological	150	gaussian	-	automatic	balanced
DLT_Gastrointestinal	400	polynomial	3	automatic	balanced
DLT_Hematological	300	gaussian	-	automatic	balanced
DLT_Nephrological	2	polynomial	20	automatic	balanced
DLT_Vascular	2	gaussian	-	automatic	balanced
DLT_Infection (Pneumonia)	1	polynomial	15	automatic	balanced
DLT_Other	3	gaussian	-	automatic	balanced
DLT_Grade	1	polynomial	10	automatic	balanced
No imaging (0=N, 1=Y)	1	gaussian	-	automatic	balanced
CR Primary	1000	gaussian	-	automatic	balanced
CR Nodal	2	polynomial	15	automatic	balanced
PR Primary	2	gaussian	-	automatic	balanced
PR Nodal	1	gaussian	-	automatic	balanced
SD Primary	10	gaussian	-	automatic	balanced
SD Nodal	3	polynomial	15	automatic	balanced
CC Regimen	10000	gaussian	-	automatic	balanced
CC modification (Y/N)	10000	gaussian	-	automatic	balanced
CR Primary 2	2	polynomial	30	automatic	balanced
CR Nodal 2	3	sigmoid	-	automatic	balanced
PR Primary 2	10000	polynomial	20	automatic	balanced
PR Nodal 2	10000	polynomial	50	automatic	balanced

TABLE I: HYPERPARAMETER DETAILS OF ALL MODELS IN THE DYNAMIC TREATMENT SIMULATOR.

Outcome	*C*	*Kernel*	*Degree*	*Gamma*	*Class Weight*
SD Primary 2	1	gaussian	-	automatic	balanced
SD Nodal 2	10	gaussian	-	automatic	balanced
DLT_Dermatological 2	100	gaussian	-	automatic	balanced
DLT_Neurological 2	10	polynomial	50	automatic	balanced
DLT_Gastrointestinal 2	30	polynomial	30	automatic	balanced
DLT_Hematological 2	100	polynomial	20	automatic	balanced
DLT_Nephrological 2	3	polynomial	3	automatic	balanced
DLT_Vascular 2	1	gaussian	-	automatic	balanced
DLT_Other 2	2	polynomial	30	automatic	balanced
Overall Survival (4 Years)	100	gaussian	-	automatic	balanced
Feeding tube 6m	10	gaussian	-	automatic	balanced
Aspiration rate Post-therapy	10	gaussian	-	automatic	balanced

5.3 Evaluation

5.3.1 Three-Step Problem Models

First of all, we test the validity of our approach by training RSMRL in the three-step problem environment presented in Section 3.1. Being this a two-decision problem, our model's architecture will have two sub-models, one for the first and one for the second decision in the sequence. The complete architecture is therefore exactly the same as that of Figure 9.

Because our ultimate goal is to use the RSMRL approach to model multiple reward functions in DTR, we focused our evaluation on environments with varying reward functions, but this is not a limitation of our approach, as it can also learn environments with varying transition probabilities.

All the three-step problems in our evaluation share one constraint: $p_1 = p_2 = p_3 = p_4 = p_5 = p_6$: the reason for this choice is that Wang et al. [2] claimed that meta-RL is particularly effective in the case of a structured environment, i.e. an environment in which the result of one action (in terms of transition or reward) gives information about the possible results of other actions (i.e. other transitions or rewards). In their case the structure referred to dependent bandits (in which the sum of the reward probabilities of each arm is always 1), while in our case the structure is given by shared transition probabilities across different junctions. The resulting "simplified" three-step task is presented in Figure 10.

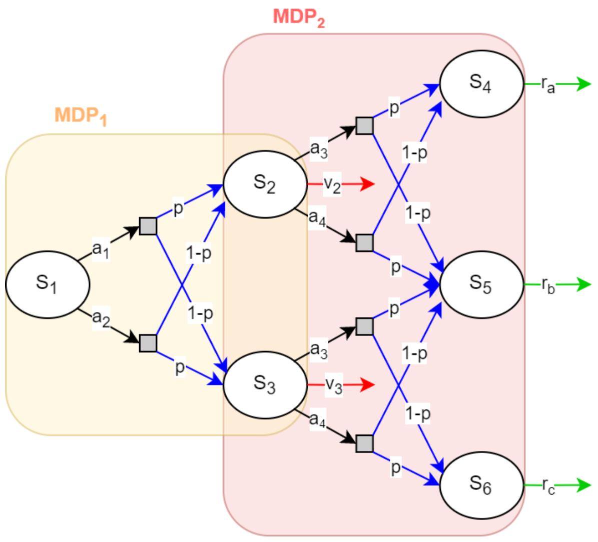


Figure 10: "Structured" three-step task: all treatment junctions share the same transition probability structure.

We train and evaluate our models on a variety of transition and reward probabilities. In particular, the transition probabilities are divided in two levels of difficulty:

- *Easy*: $p = 0.9$.
- *Hard*: $p = 0.75$.

The "difficulty" of the transitions refers to the difficulty of the model to learn from its surroundings: the closer p and $1 - p$ are, the harder it is for the model to learn the transition probabilities, and therefore to choose the action with higher probability of leading to rewards.

Analogously, the (varying) reward probabilities are divided into three levels of difficulty:

- *Easy*: one of $\{r_a, r_b, r_c\}$ is assigned reward probability 0.9, while the other two are assigned reward probability 0.05.
- *Medium*: one of $\{r_a, r_b, r_c\}$ is assigned reward probability 0.75, while the other two are assigned reward probability 0.125.
- *Hard*: one of $\{r_a, r_b, r_c\}$ is assigned reward probability 0.6, while the other two are assigned reward probability 0.2.

Once again, the concept of difficulty corresponds to the difficulty for the model to learn the dynamics of the system and to act optimally (in this case, identify the state with the highest reward probability).

A different RSMRL model was trained for all transition-reward difficulty combination.

The standard RL model used in our RSMRL models is the Advantage Actor Critic (A2C) [31,32] (although our approach, as meta-RL, can employ any RL algorithm). The main elements of the A2C loss functions are *entropy*, *value loss*, and *policy loss*.

The *entropy* of an episode of the jth model is defined as the sum of the entropies of each trial in the episode, $H(\pi_t^j)$. The entropy of a trial is computed as:

$$H(\pi_t^j(\dot{|}x_t)) = -\sum_{a \in \mathcal{A}} \pi_t^j(a|x_t) log \pi_t^j(a|x_t) \tag{5.2}$$

Entropy measures how “sure” a policy is: if the probabilities of all actions in the policy are similar then entropy will be high, otherwise if one action has a much higher policy probability than the others then entropy will be low. Considering the entropy as part of the loss function is called *entropy regularization*, and is used to encourage exploration, to increase the chances of finding the global optimum.

The *value loss* of an episode of the jth model is the sum of the value losses of each trial t of the jth model. The value loss of a trial is computed as the squared difference between the reward obtained at trial t, r_t^j, and the value of the state computed by the model at time t, $V_t^j(x_t^j)$. It measures how far off the state value estimated by the model is from the actual reward obtained.

$$ValueLoss_t^j = (r_t^j - V_t^j(x_t^j))^2 \tag{5.3}$$

The *policy loss* of an episode of the jth model is the sum of the policy losses of each trial in the episode. The policy loss of a single trial is computed as:

$$PolicyLoss_t^j = -log(\pi(a_t|x_t))(r_t^j + \gamma V(x_{t+1}^j) - V(x_t^j)) \tag{5.4}$$

The *loss* of an episode is then computed as:

$$Loss^j = \sum_{t=1}^{n_t} PolicyLoss_t^j + ValueLoss_t^j - H(\pi_t^j(x_t^j)) \tag{5.5}$$

The RNN of the model is a Long Short-Term Memory (LSTM) [33] with hidden state size 20 (again, any type of RNN can be employed). The training consisted of 40,000 episodes of 100 trials each. For training we used Adam Optimizer [34] with learning rate 0.001.

Evaluation was done on 1000 episodes of 100 trials each. We report rewards and cumulative rewards of each trial, averaged across episodes. Each model is evaluated in all environments with the same transition difficulty as the environment it was trained on, but with possibly different reward difficulties: for example, the model trained on the easy transition-easy reward environment is evaluated on environments with easy transitions and easy, medium, and hard rewards. This way, we can evaluate the generalization capabilities of the models to previously unseen environments that share some regularities with the ones they were trained on.

We compare these rewards to those obtained by a random choice of action as well as the theoretically optimal achievable reward.

5.3.2 Dynamic Treatment of Oropharyngeal Squamous Cell Carcinoma with Multiple Rewards

After evaluating the RSMRL approach on the "simple" three-step problem, we now move on to training a model on the MDACC OPC dataset presented in Section 4.1. The environment on which we train and test our model is the Dynamic Treatment Simulator (DTS) presented in 5.2. Because this is a real-life problem, we cannot change the transition probabilities given by the DTS; we can only change the reward distribution.

In order to model varying preferences over outcomes we define a *preference vector* $\vec{w}$ analogous to that proposed by Barrett et al. [17] and Lizotte et al. [18, 19]: the final reward is then given as the dot product of the preference vector and the rewards vector. Unlike in Barret et al. and Lizotte et al., this linear approximation of the preferences is not a limitation of our approach: any function (linear or not) of the final outcomes can be used to model varying preferences, and we just chose to start our evaluation from the simplest (and most realistic) linear combination.

The reward variables, as mentioned in Section 2.2.1 and 4.1, are Overall Survival at 4 years (after the completion of the treatment sequence) and Dysphagia (more precisely, we want to maximize the absence of dysphagia symptoms). These outcomes are often at odds with each other, as previously mentioned, because maximizing survival often requires a very aggressive chemo-radiotherapeutic treatment, which is likely to cause the high level of toxicities that lead to Dysphagia: we therefore expect that different reward vectors will lead to very different reward functions, and, as a consequence, very different optimal policies.

We trained one RSMRL model for each of the four following preference difficulties:

- *Easy*: one outcome is randomly chosen to be assigned preference weight 0.9, the other 0.1.
- *Medium*: one outcome is randomly chosen to be assigned preference weight 0.75, the other 0.25.
- *Hard*: one outcome is randomly chosen to be assigned preference weight 0.6, the other 0.4.
- *Uniform*: one outcome's preference weight w_0 is randomly sampled from a uniform distribution, the other is $1 - w_0$

Here again the notion of difficulty is related to the difficulty for the model to learn the optimal policy: the closer the two reward weights are, the harder it is for the model to learn which outcome is preferred.

Compared to the difficulties presented for the three-step problem in 5.3.1, we introduced a new one, the *uniform* difficulty: this is the difficulty level that can truly handle any patient's preference, seen or unseen. This preference environment is clearly the most general, and therefore the most realistic, of the four, and as such is the ultimate goal of our approach.

Our models are again composed of A2C and LSTM, but this time the LSTM has a hidden state size of 48: this larger state size compared to the three-step problem models is chosen for the following reasons:

- The *state space representation is much larger* than the 1-2 states of the three-step problem: we have a large number of medical features (see Appendix B), so a smaller hidden state might not be able to embed all the information necessary for the model to act optimally.
- The *problem is more complex*: while the three-step problem has a very simple and repetitive structure, the transitions of dynamic treatment are clearly less regular and predictable. Therefore more information may be required to adequately represent the hidden state.

The training consisted of 20,000 episodes of 100 trials each. In each trial, a random patient is sampled from the training set and the outcomes of the treatment prescribed by the RSMRL model are simulated with the DTS. For training we used Adam Optimizer [34] with learning rate 0.001 (same as for the three-step problem).

Evaluation was performed on a separate set of patients, over 1000 episodes of 100 trials. In this case each of the four models is not only evaluated on the environment (i.e. preference difficulty) in which it was trained, but on all four difficulties, to test generalization properties of the models. We report the average cumulative reward per episode, as compared to random performance. Because transition (and reward) probabilities are unknown, we cannot compare the RSMRL performance to the theoretically optimal performance, therefore our only measure is the improvement over random chance performance.

CHAPTER 6

RESULTS

6.1 The Three-Step Problem

6.1.1 Training Performance

First of all, we evaluate the training performance of the RSMRL models to address the concerns about error propagation and training delays that we mentioned in 5.1.1.

Figure 11 shows the rewards, values, entropies, and losses in training of the hard transitions, easy rewards three-step problem RSMRL model, both for the first and second decision.

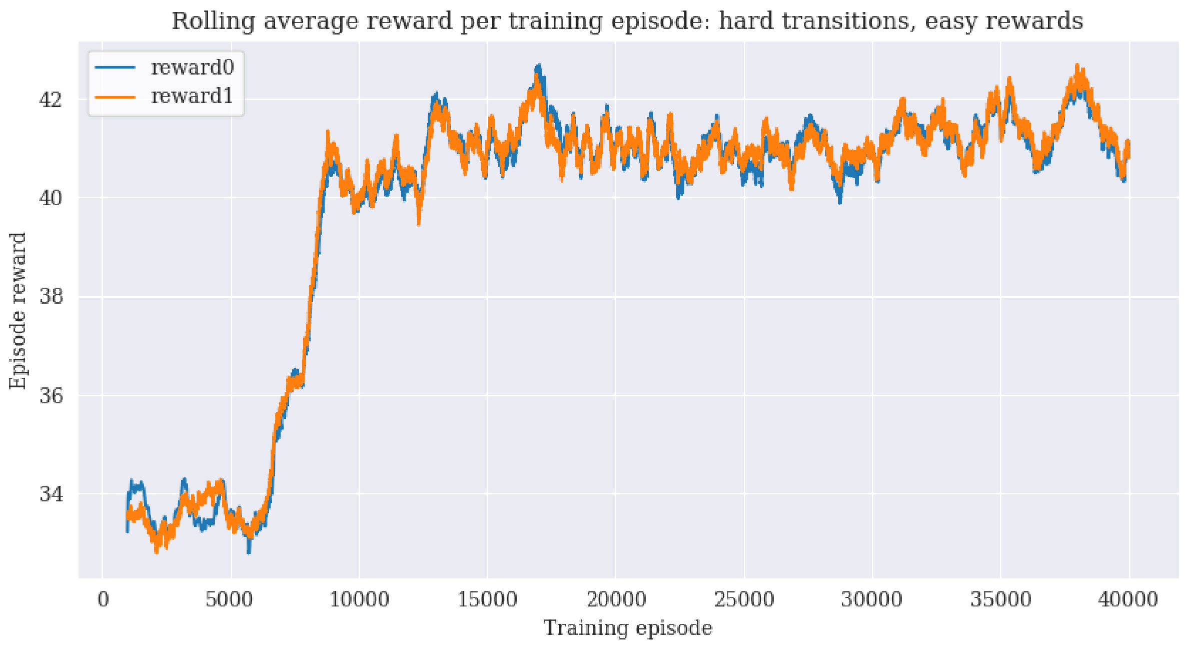
The *value* of a trial t of the jth decision is the value $V_t^j(x_t^j)$ outputted by the jth model at trial t.

We see that the trends of the first decision model (which is trained on the values propagated by the second decision model) closely (but not exactly) follow the trends of the second decision model.

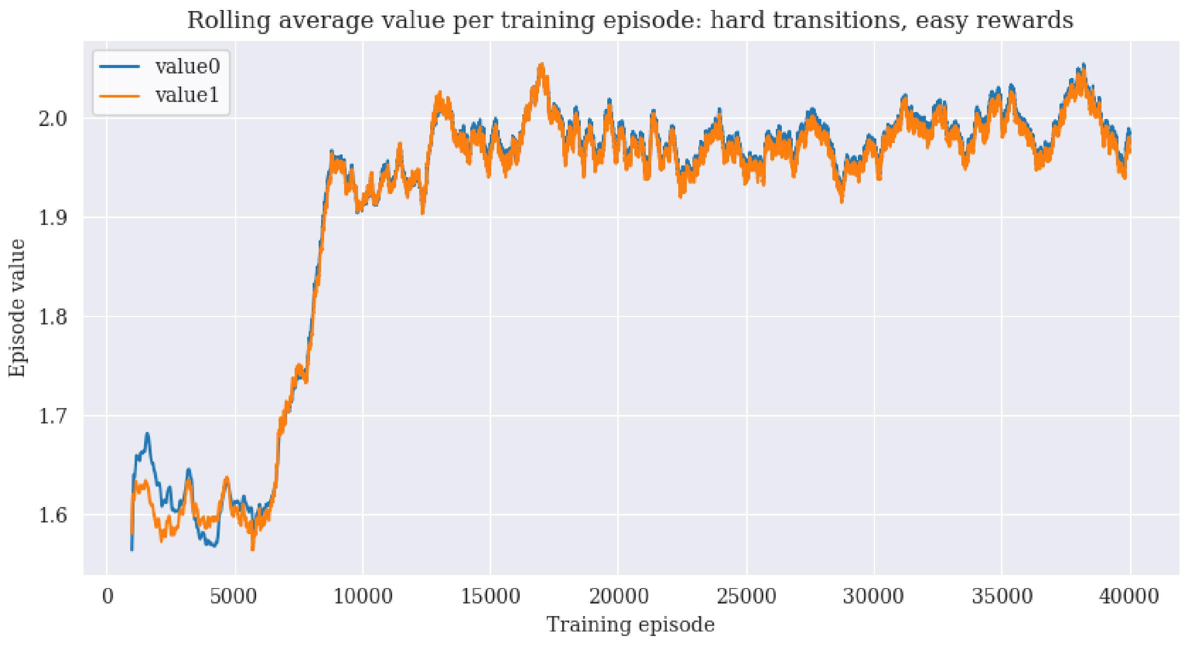
This not only dissipates the concerns about divergence or convergence rate of training the complete model sequence all at the same time, but also proves that the *value feedback* is an appropriate approximation of the final reward for intermediate decisions. The propagated value gives enough information about the outcome, but still filters out the "responsiblity" of other models in achieving said outcomes, as we can see by the fact that the trends of the first decision model are significantly smoother than those of the second decision model.

Moreover, if we look at the reward plot (Figure 11a) we see that it takes a relatively short time (about 10,000 training episodes) for the model to learn the dynamics of the problem and reach convergence, with a remarkably steep learning curve between episode 5,000 and 10,000. This is especially notable considering that the considered model is trained on an environment with hard transition probabilities, so it's one of the hardest environments to learn.

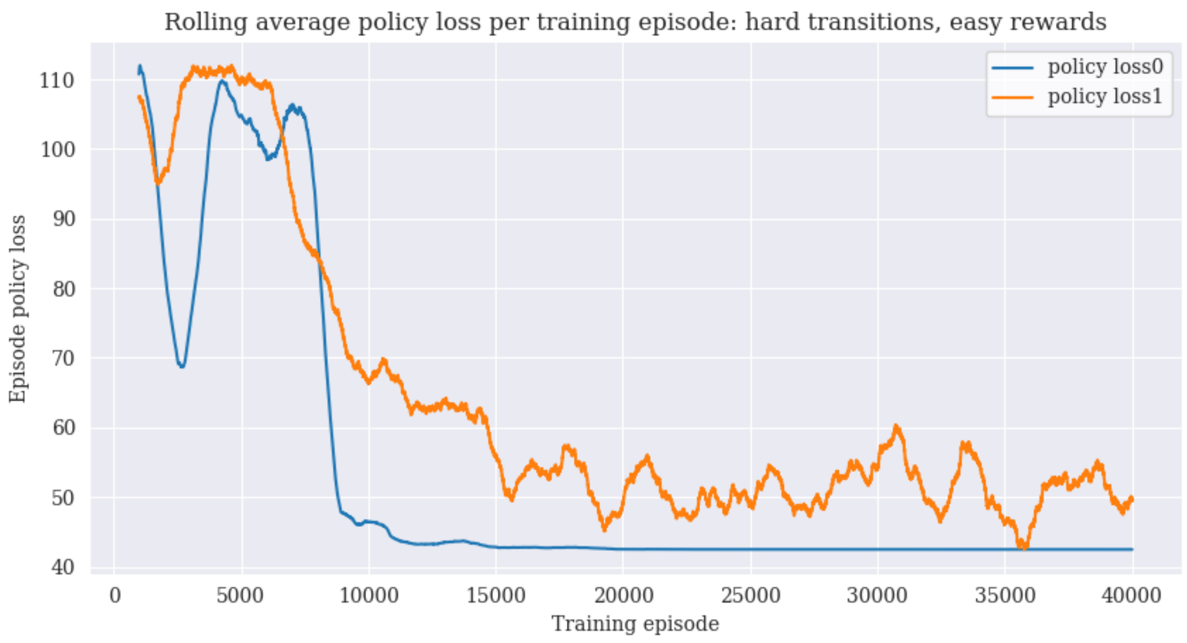
Sequential models are therefore capable of training together and quickly propagate accurate information about the environment even before reaching convergence, leading to relatively short training times to reach convergence.



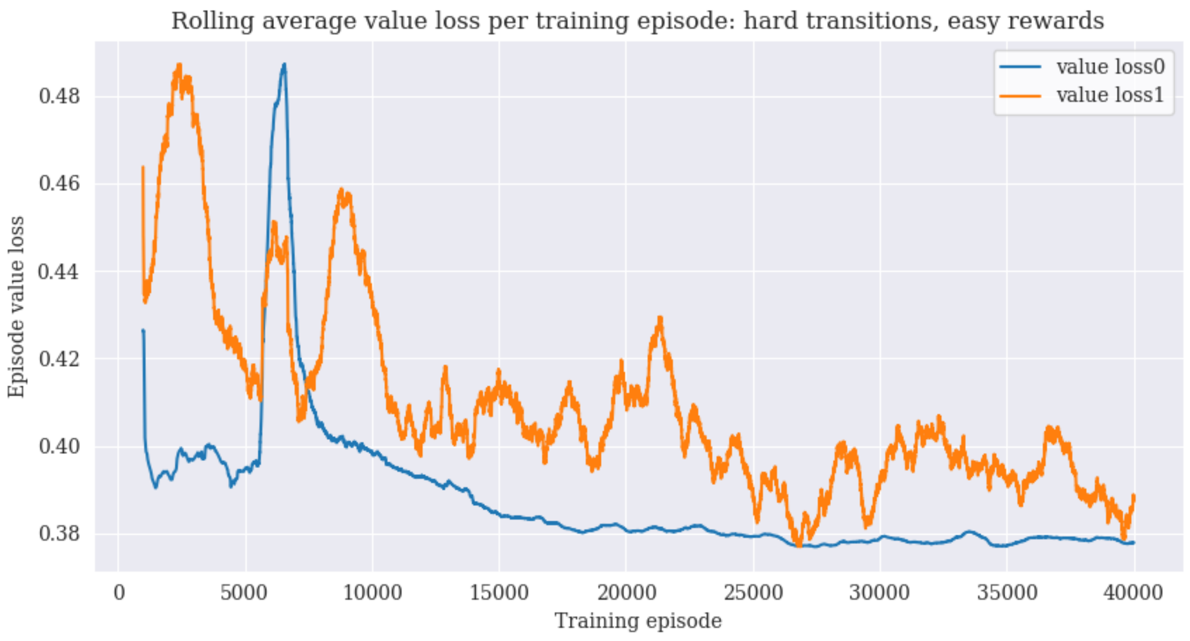
(a) Rolling average of rewards per training episode, for decision 1 (*reward0*) and 2 (*reward1*)



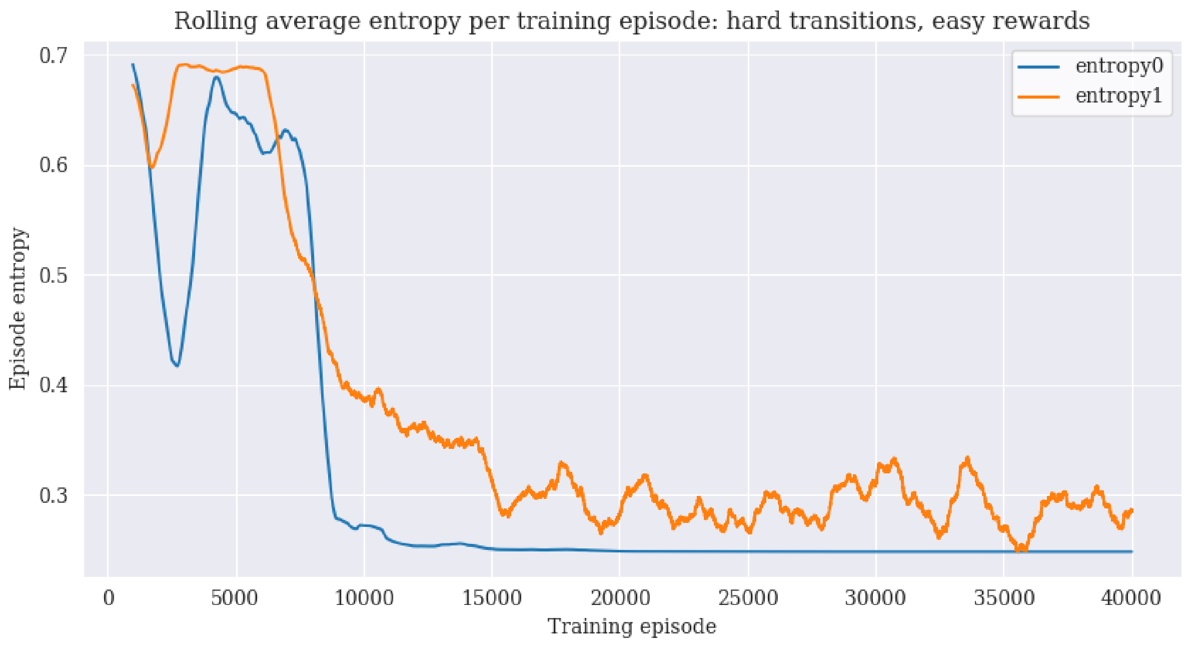
(b) Rolling average of episode values per training episode, for decision 1 (*value0*) and 2 (*value1*)



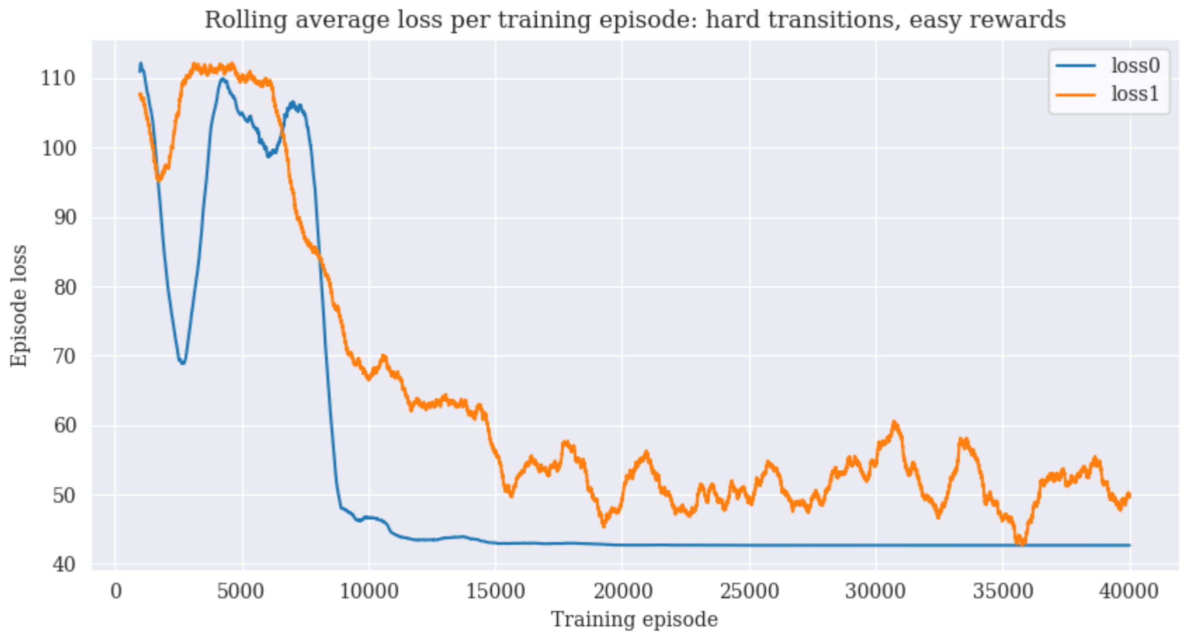
(c) Rolling average of policy loss per training episode, for decision 1 (*policy loss0*) and 2 (*policy loss1*)



(d) Rolling average of value loss per training episode, for decision 1 (*value loss0*) and 2 (*value loss1*)



(e) Rolling average of entropy per training episode, for decision 1 (*entropy0*) and 2 (*entropy1*)



(f) Rolling average of loss per training episode, for decision 1 (*loss0*) and 2 (*loss1*)

Figure 11: Training performance of the hard transitions, easy rewards three-step problem RSMRL model, for each of the two decisions.

6.1.2 Testing performance

Now that we have validated our choice to train all models in the sequence at the same time, we can analyze their performance in the evaluation.

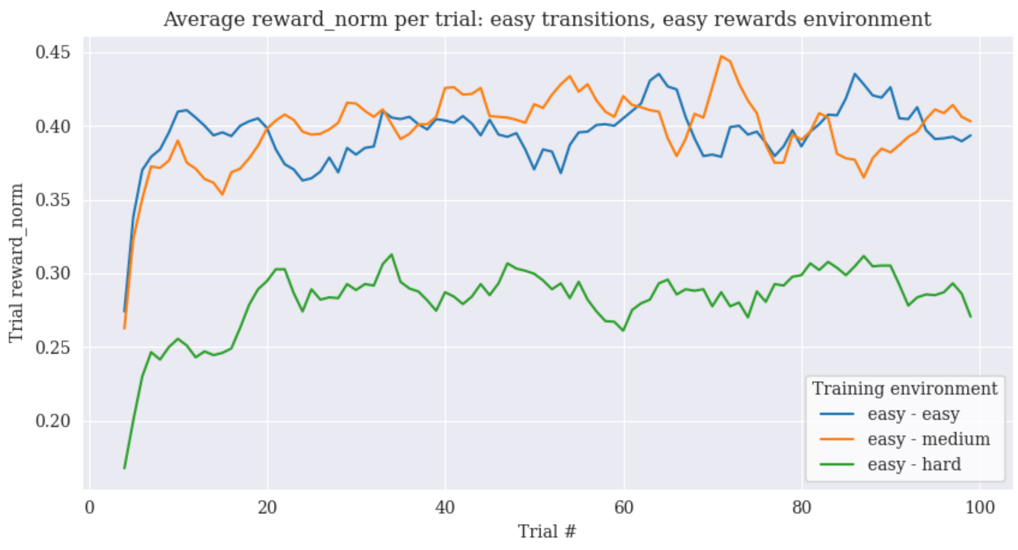
Firstly, in Figure 12 we show the performance in each transition-reward difficulty combination of all the models trained with the same transition difficulty (but possibly a different reward difficulty). Rewards are normalized so that random choice has reward 0, and optimal choice has reward 1.

These plots clearly show (especially in the easier environments) the learning curve of the model in each evaluation episode: in just a few trials, the models are able to build a strong belief on the dynamics of the environment they are currently in, and adapt their policy accordingly. As is to be expected, the performance in more complex environments shows a higher variance and a slower learning curve, as it is harder for the models to assess their surroundings. However, even in the hardest environments the models manage to improve on random chance performance, so they are able to capture some structure of the environment.

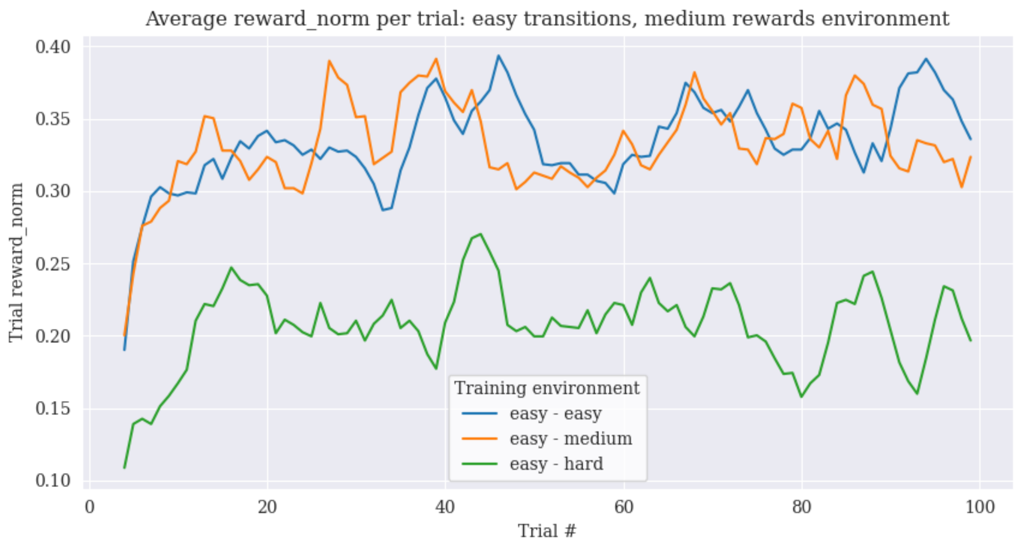
More interestingly, these results show that models trained on easier environments tend to generalize better to unseen circumstances: both in the easy and in the hard transition environments, the best-performing model in terms of learning curve and variance is the model trained on the easy rewards environments, regardless of the reward difficulty of the environment it was evaluated on, outperforming even the models trained on the same environment. This confirms the claim of Wang et al. [2] that training meta-RL models on easier tasks can actually lead to better performance on any given environment than training a model for that specific environment.

Table II reports the average cumulative reward per episode of all RSMRL models, in all training and testing environments, compared to the average cumulative reward obtained by random choice of action. These values show the consistent improvement of RSMRL models over random performance, and further demonstrate the superiority of models trained on easier environments in terms of performance and adaptability.

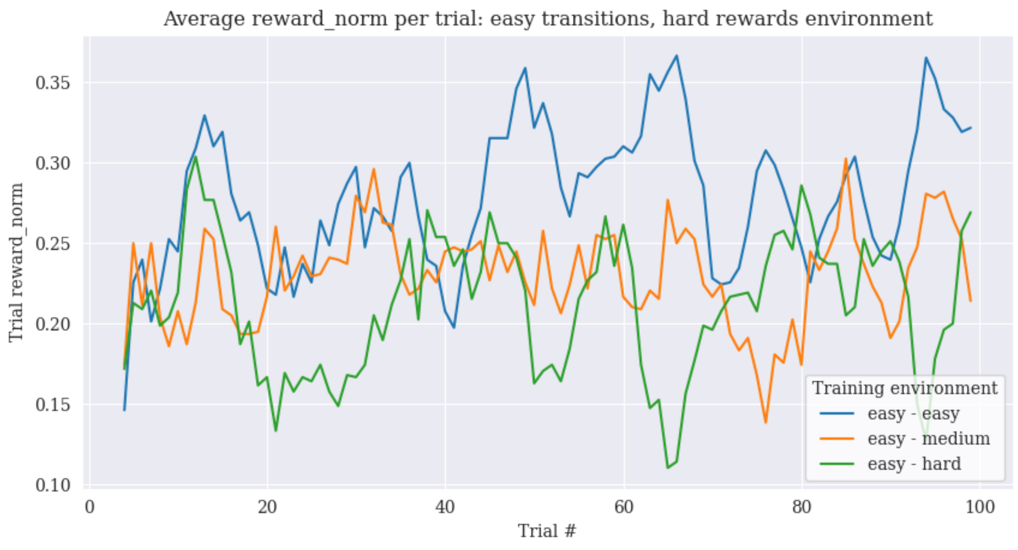
Overall, Figure 12 and Table II show that the three-step problem RSMRL models are capable, without further training, to adapt to any environment, as do meta-RL models in single-step settings. This once again proves the validity of the *state and value feedbacks.* Moreover, models trained on simpler environments lead to higher rewards and lower variance on any environment, when compared to models trained in more complex settings. Training on easier tasks therefore improves performance and adaptability of the models.



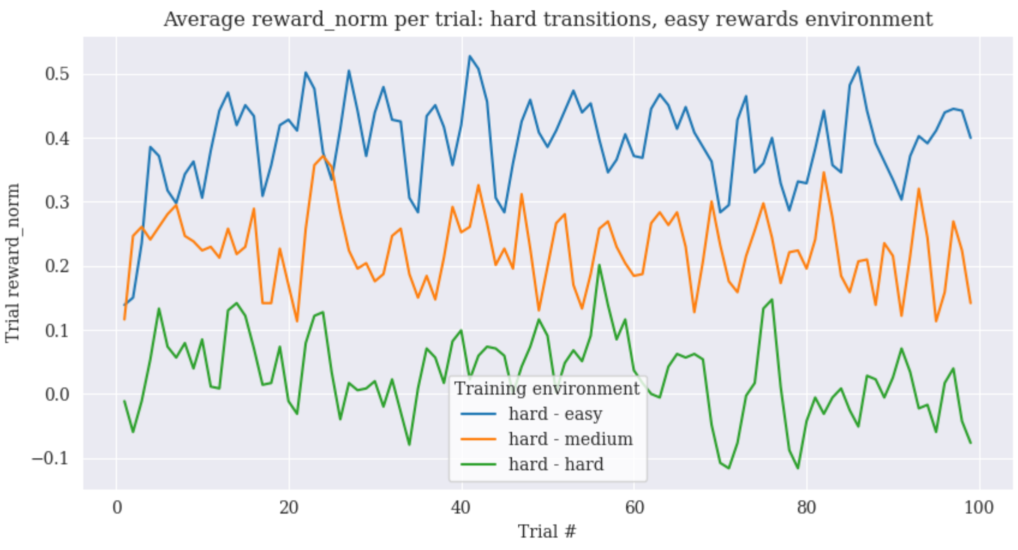
(a) Average normalized evaluation reward per trial in the easy transitions, easy rewards three-step problem environment, of all the RSMRL models trained in the easy transitions environment



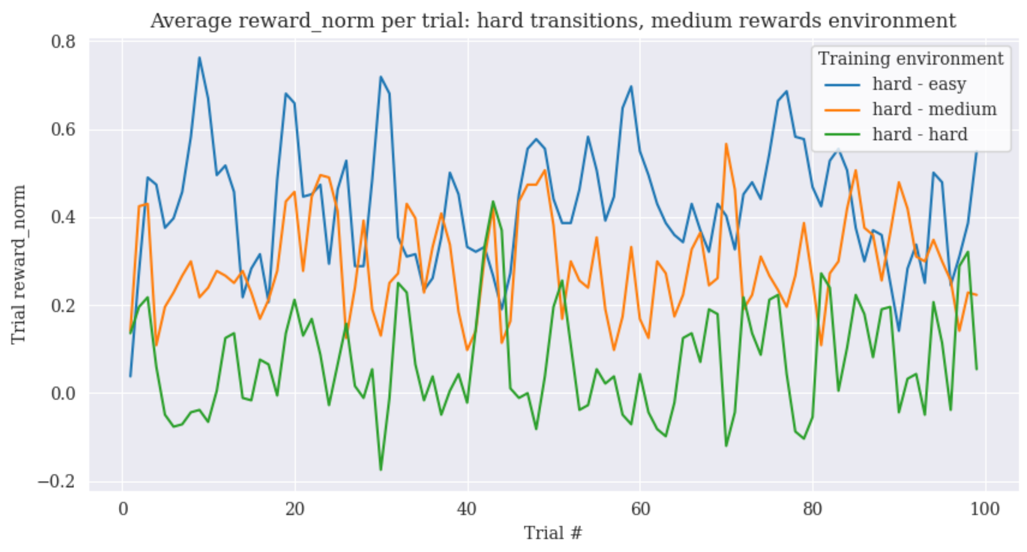
(b) Average normalized evaluation reward per trial in the easy transitions, medium rewards three-step problem environment, of all the RSMRL models trained in the easy transitions environment



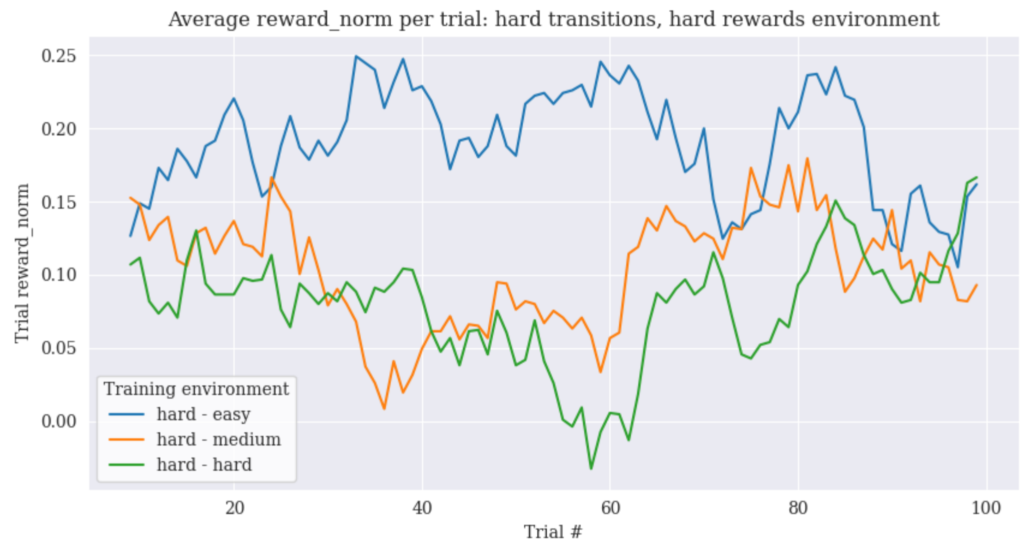
(c) Average normalized evaluation reward per trial in the easy transitions, hard rewards three-step problem environment, of all the RSMRL models trained in the easy transitions environment



(d) Average normalized evaluation reward per trial in the hard transitions, easy rewards three-step problem environment, of all the RSMRL models trained in the hard transitions environment



(e) Average normalized evaluation reward per trial in the hard transitions, medium rewards three-step problem environment, of all the RSMRL models trained in the hard transitions environment



(f) Average normalized evaluation reward per trial in the hard transitions, hard rewards three-step problem environment, of all the RSMRL models trained in the hard transitions environment

Figure 12: Average reward per evaluation trial in all transition-reward difficulty combinations of three-step problem environments, of all the models trained on the same transition difficulty. Rewards are normalized so that random chance performance is 0, and optimal performance is 1.

TABLE II: AVERAGE CUMULATIVE REWARD PER EVALUATION EPISODE OF RANDOM CHOICE AND RSMRL MODELS, FOR ALL TRAINING AND TESTING ENVIRONMENTS.

Transition difficulty	*Reward difficulty (training)*	*Reward difficulty (environment)*	*Random*	*RSMRL*	*Improvement*
easy	easy	easy	33.57	48.59	44.74%
		medium	33.42	42.15	26.13%
		hard	33.28	37.32	12.13%
	medium	easy	33.41	48.7	45.8%
		medium	33.1	42.1	27.16%
		hard	33.01	36.53	10.68%
	hard	easy	32.95	44.11	33.85%
		medium	32.81	38.65	17.83%
		hard	33.25	36.26	9.05%
hard	easy	easy	33.08	39.88	20.58%
		medium	33.71	36.93	9.55%
		hard	33.12	35.01	5.69%
	medium	easy	33.13	36.95	11.54%
		medium	33.74	35.66	5.7%
		hard	33.35	34.13	2.34%
	hard	easy	33.52	33.52	0.01%
		medium	33.28	33.67	1.18%
		hard	33.77	33.91	0.43%

6.2 The Dynamic Treatment of Oropharyngeal Squamous Cell Carcinoma with Multiple Rewards Problem

6.2.1 Dynamic Treatment Simulator

The prediction accuracy of the DTS with 95% confidence intervals was assessed with out-of-bag evaluation of 1000 models trained on stratified bootstrapped samples. The resulting accuracy (and 95% CI) is presented in Table III. The average bootstrapped prediction accuracy of the individual DTS models was 87.35%, and median accuracy was 92.07%.

We also evaluated the prediction accuracy at the whole policy level, meaning that the DTS predicts each outcome of the sequence from start to finish, and we compare the final predicted outcome to the real one. The average prediction accuracy on the test set outcomes is 83.21%, with 83.96% accuracy for Overall Survival, and 82.46% for Dysphagia (88.43% for Feeding Tube, and 83.58% for Aspiration Rate).

TABLE III: PREDICTION ACCURACY (WITH 95% CONFIDENCE INTERVALS IN BRACKETS) OF THE STRATIFIED BOOTSTRAPPED EVALUATION OF THE DYNAMIC TREATMENT SIMULATOR.

Outcome	*Accuracy (CI)*
Prescribed Chemo	83.0% (77.32%, 87.57%)
Chemo Modification (Y/N)	82.09% (76.96%, 86.34%)
Dose modified	92.39% (89.23%, 94.95%)
Dose delayed	92.39% (89.12%, 95.17%)
Dose cancelled	91.58% (87.68%, 94.77%)
Regimen modification	93.54% (84.36%, 95.88%)
DLT (Y/N)	81.51% (77.25%, 85.42%)
DLT_Dermatological	92.77% (23.95%, 95.29%)
DLT_Neurological	92.17% (88.66%, 95.1%)
DLT_Gastrointestinal	89.6% (85.86%, 92.96%)
DLT_Hematological	90.1% (86.17%, 93.23%)
DLT_Nephrological	99.03% (98.0%, 100.0%)
DLT_Vascular	98.45% (96.45%, 100.0%)
DLT_Infection (Pneumonia)	98.98% (94.42%, 100.0%)

TABLE III: PREDICTION ACCURACY (WITH 95% CONFIDENCE INTERVALS IN BRACKETS) OF THE STRATIFIED BOOTSTRAPPED EVALUATION OF THE DYNAMIC TREATMENT SIMULATOR.

Outcome	*Accuracy (CI)*
DLT_Other	95.08% (90.82%, 97.57%)
DLT_Grade	73.85% (53.84%, 79.9%)
No imaging (0=N, 1=Y)	100.0% (100.0%, 100.0%)
CR Primary	83.51% (78.82%, 87.56%)
CR Nodal	94.79% (90.5%, 97.03%)
PR Primary	81.47% (76.84%, 86.27%)
PR Nodal	92.93% (90.0%, 95.65%)
SD Primary	95.1% (91.96%, 97.84%)
SD Nodal	96.58% (94.47%, 98.05%)
CC Regimen	70.0% (64.68%, 75.27%)
CC modification (Y/N)	70.53% (64.92%, 76.06%)
CR Primary 2	79.22% (23.03%, 85.22%)
CR Nodal 2	55.5% (49.01%, 61.54%)
PR Primary 2	78.92% (74.26%, 83.25%)

TABLE III: PREDICTION ACCURACY (WITH 95% CONFIDENCE INTERVALS IN BRACKETS) OF THE STRATIFIED BOOTSTRAPPED EVALUATION OF THE DYNAMIC TREATMENT SIMULATOR.

Outcome	*Accuracy (CI)*
PR Nodal 2	52.5% (46.19%, 58.03%)
SD Primary 2	99.48% (98.46%, 100.0%)
SD Nodal 2	96.5% (94.12%, 98.04%)
DLT_Dermatological 2	91.99% (87.63%, 95.17%)
DLT_Neurological 2	95.79% (5.96%, 97.46%)
DLT_Gastrointestinal 2	89.74% (85.22%, 93.65%)
DLT_Hematological 2	92.71% (89.42%, 95.16%)
DLT_Nephrological 2	92.25% (88.17%, 97.94%)
DLT_Vascular 2	100.0% (99.45%, 100.0%)
DLT_Other 2	93.97% (89.73%, 96.86%)
Overall Survival (4 Years)	78.23% (73.2%, 82.92%)
Feeding tube 6m	74.37% (68.81%, 79.4%)
Aspiration rate Post-therapy	75.0% (69.38%, 80.0%)

6.2.2 Recurive Sequential Meta-Reinforcement Learning for Dynamic Treatment of Oropharyngeal Squamous Cell Carcinomas

6.2.2.1 Training performance

In Figure 13 we report the rolling average of rewards per episode during training of all the four RSMRL for OPC models: we note that, compared to the training performance of the three-step problem, here the learning curve is slightly more complex. Despite that, all models reach convergence in the first 10,000 training episodes, same as the three-step problem models.

Once again we note the superiority in performance of models trained on simpler environments: while the model trained on the hard preferences environment still shows a learning curve, it is significantly lower in performance at convergence when compared to the others.

We also see that the uniform preferences model (our ultimate goal in terms of environment) is the slowest to reach convergence, as is expected given the vastness of the distribution of environments it is trained on when compared to the other distributions. However, at convergence, it reaches the same performance as the easy and medium models, which are trained in much simpler environments.

Overall, we can say that the learning curve in training shows that RSMRL can relatively quickly learn the peculiarities even of rather complex environments, and therefore seems to be suitable for realistic sequential decision-making scenarios.

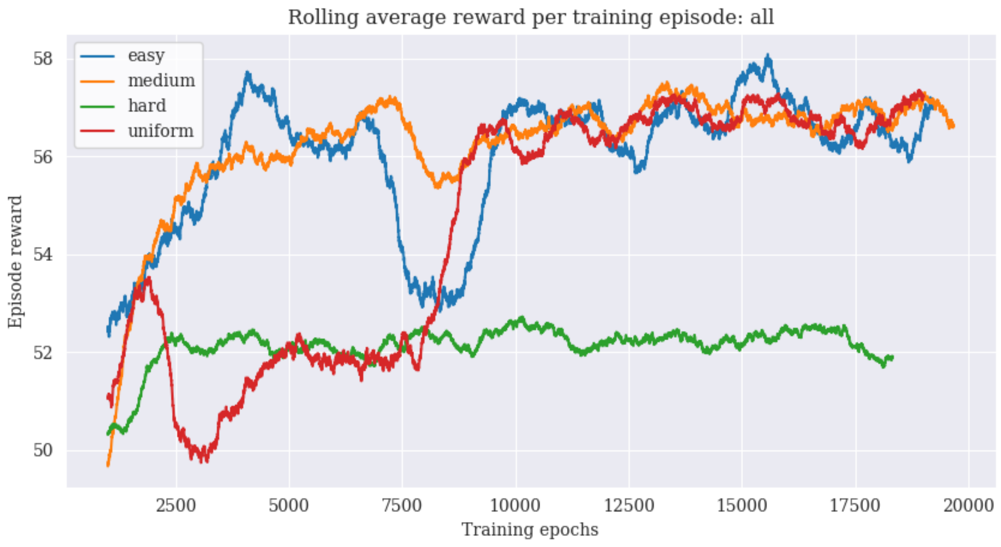


Figure 13: Rolling average of rewards per training episode, of all the RSMRL for OPC models.

6.2.2.2 Testing Performance

We now report in Table IV the average cumulative reward in the evaluation of all the four models, each tested in all four environments. Their performance is compared to that of random choice, reporting the relative improvement.

As observed in the training performance in Section 6.2.2.1 and in the performance of the three-step problem RSMRL models in Section 6.1, we confirm that the models trained on the simplest environments mostly outperform the models of the more complex environments on all environments, with the easy and medium models greatly outperforming the hard model.

However, in this case, the model trained on the uniform environment, which is undoubtedly the most complex, achieves slightly lower but still good performance compared to the easy and medium ones, and, more importantly, is the best-performing one in the uniform testing environment.

Since the ultimate goal of applying RSMRL to DTR is to be able to model any set of patient preferences (i.e. the uniform preferences environment), we must ultimately recognise the uniform environment model as the best-performing one, as it is the one that most of all displays the ability to adapt to any given set of preferences.

We have therefore showed that RSMRL is capable of capturing the dynamics even of a complex real-life sequential decision-making problem such as the DTR of OPC, and provides a model capable of rapidly adapting even to complex distributions of preferences over outcomes.

TABLE IV: AVERAGE CUMULATIVE REWARDS OF EACH TRAINING-TESTING ENVIRONMENT COMBINATION OF MODELS, COMPARED TO RANDOM CHOICE PERFORMANCE.

Model difficulty (training)	*Environment difficulty (testing)*	*Random*	*RSMRL*	*Improvement*
easy	easy	49.3	58.61	18.88%
	medium	49.02	57.04	16.36%
	hard	48.79	56.43	15.66%
	uniform	50.82	55.95	10.1%
medium	easy	49.3	58.2	18.06%
	medium	49.02	58.23	18.78%
	hard	48.79	57.13	17.09%
	uniform	50.82	56.7	11.57%
hard	easy	49.3	50.16	1.76%
	medium	49.02	51.98	6.04%
	hard	48.79	51.5	5.55%
	uniform	50.82	50.87	0.11%
uniform	easy	49.3	57.38	16.4%
	medium	49.02	56.45	15.15%
	hard	48.79	56.31	15.42%
	uniform	50.82	56.85	11.87%

CHAPTER 7

CONCLUSION

In this work, we highlighted the main shortcomings of the state-of-the-art deep meta-Reinforcement Learning (meta-RL) approaches, in particular the strict dependence on including single-step rewards in order for the meta-learning to work.

We suggested the potential of adapting these approaches to handle fixed-sequence series of decisions in the absence of intermediate reward, and the implications this would have in the computational healthcare field of Dynamic Treatment Regimes (DTR) to train models capable of handling any previously unseen interpolation of multiple reward functions, such as varying patient-specific preferences on the outcome of a therapeutic course.

We proposed a new approach, Recursive Sequential deep Meta-Reinforcement Learning (RSMRL), that extends the applicability of meta-RL to fixed-length multi-step problems with no intermediate rewards. An RSMRL model is composed of a series of meta-RL models, one for each decision in the sequence, and incorporates two new types of feedback: the *state feedback* and the *value feedback*. The *state feedback* is added to the action and reward feedback to provide more accurate insight on the results of an action in the current environment, giving the model a deeper understanding of transition probabilities. The *value feedback* substitutes the reward feedback in the intermediate-decisions models, and compensates for the absence of intermediate-step rewards by including in the input the value of the state that the previous action led to, as computed by the following model in the sequence. It is this *model-to-model feedback* that introduces recursion in our approach, making it possible to propagate appropriately filtered information on the final reward back to previous decisions in the sequence, so as to encode the partial contributions of each decision value to the final outcome.

We first evaluate our approach on a synthetic environment, the three-step problem, consisting of a two-decisions sequence, with reward being observed only after the second decision. Our results on the training performance show that the models of all the decisions in the sequence are capable of learning synchronously, without significant delays or divergences in performance between the first decision and second decision model, thus confirming the validity of the *value feedback* to propagate reward information back to intermediate decisions. The evaluation reward performance showed the learning curve of which trained RSMRL models are capable when applied to previously unseen environment, and the average cumulative reward per episode showed a consistent improvement over random choice performance, as well as the higher ability of models trained in easier environments to adapt to new more complex scenarios, even outperforming models trained for those exact environments.

Finally, we applied the RSMRL to a real-life DTR problem of a three-step sequential treatment of Oropharyngeal Squamous cell Carcinomas (OPC), based on a dataset of real patients collected at the MD Anderson Cancer Center (MDACC). After building a Dynamic Treatment Simulator to emulate the treatment environment, we trained multiple RSMRL models in this environment by modeling different patient-specific preferences over treatment outcomes as different reward functions. The training performance revealed the same trends seen in the three-step problem models, proving the ability of RSMRL to capture the dynamics even of complex real-life environments. The evaluation performance showed once again consistent improvement over random choice performance, thus obtaining models capable of handling any type of patient preference weights, modeled as a uniform distribution.

In the future, we hope to further improve the applicability of RSMRL by testing its performance on environments with possibly varying transition probabilities, in addition to the varying reward probabilities evaluated in this work.

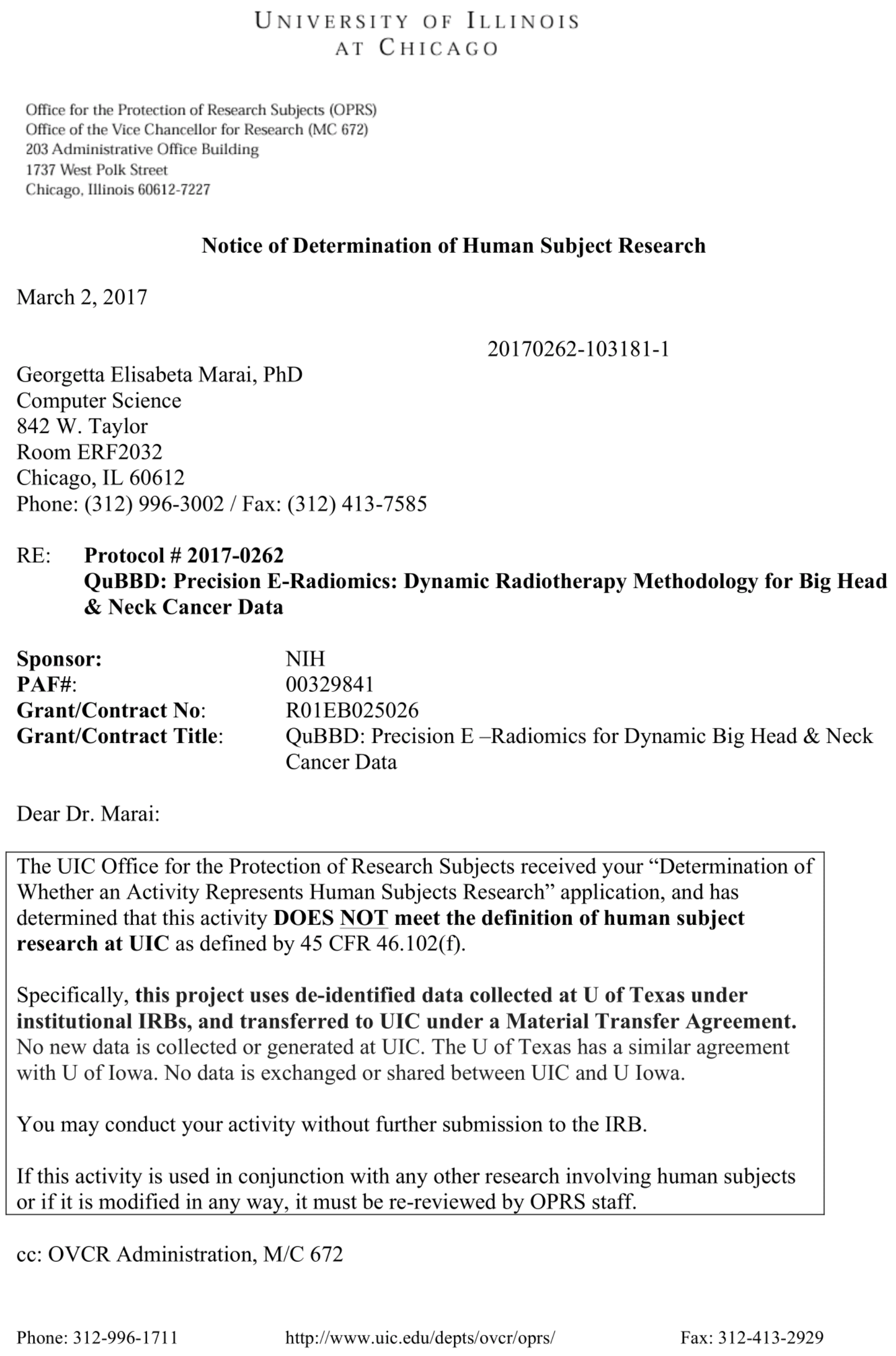
We also hope to evaluate its application to the DTR with multiple reward functions setting in the case of non-linear preference functions, and possibly in the presence of a varying number of unseen reward functions rather than a fixed one.

APPENDICES

Appendix A

HUMAN SUBJECT RESEARCH DETERMINATION

Appendix A (continued)



UNIVERSITY OF ILLINOIS
AT CHICAGO

Office for the Protection of Research Subjects (OPRS)
Office of the Vice Chancellor for Research (MC 672)
203 Administrative Office Building
1737 West Polk Street
Chicago, Illinois 60612-7227

Notice of Determination of Human Subject Research

March 2, 2017

20170262-103181-1

Georgetta Elisabeta Marai, PhD
Computer Science
842 W. Taylor
Room ERF2032
Chicago, IL 60612
Phone: (312) 996-3002 / Fax: (312) 413-7585

RE: **Protocol # 2017-0262**
QuBBD: Precision E-Radiomics: Dynamic Radiotherapy Methodology for Big Head & Neck Cancer Data

Sponsor:	NIH
PAF#:	00329841
Grant/Contract No:	R01EB025026
Grant/Contract Title:	QuBBD: Precision E –Radiomics for Dynamic Big Head & Neck Cancer Data

Dear Dr. Marai:

The UIC Office for the Protection of Research Subjects received your "Determination of Whether an Activity Represents Human Subjects Research" application, and has determined that this activity **DOES NOT meet the definition of human subject research at UIC** as defined by 45 CFR 46.102(f).

Specifically, **this project uses de-identified data collected at U of Texas under institutional IRBs, and transferred to UIC under a Material Transfer Agreement.** No new data is collected or generated at UIC. The U of Texas has a similar agreement with U of Iowa. No data is exchanged or shared between UIC and U Iowa.

You may conduct your activity without further submission to the IRB.

If this activity is used in conjunction with any other research involving human subjects or if it is modified in any way, it must be re-reviewed by OPRS staff.

cc: OVCR Administration, M/C 672

Phone: 312-996-1711 http://www.uic.edu/depts/ovcr/oprs/ Fax: 312-413-2929

Appendix B

DEMOGRAPHICS OF MDACC OPC DATASET

TABLE V: MAIN FEATURE DEMOGRAPHICS OF THE MDACC OPC DATASET FOR EACH TREATMENT JUNCTION.

Characteristic	*All Patients (536)*	*Training set (402)*	*Testing set (134)*
Pre-treatment variables (before D1)			
Age at Diagnosis (Calculated)			
Mean (SD)	58.9 (9.5)	58.5 (9.4)	60.2 (9.6)
Pathological Grade			
I	6 (1.4%)	2 (0.6%)	4 (3.5%)
II	154 (35.2%)	114 (35.3%)	40 (35.1%)
III	274 (62.7%)	206 (63.8%)	88 (59.6%)
IV	3 (0.7%)	1 (0.3%)	2 (1.8%)
N/A	99	79	20
Gender			
Female	65 (12.1%)	47 (11.7%)	18 (13.4%)
Male	471 (87.9%)	355 (88.3%)	116 (86.6%)
HPV/P16 status			
Negative	43 (8%)	33 (8.2%)	10 (7.5%)
Positive	305 (56.9%)	228 (56.7%)	77 (57.5%)
Unknown	188 (35.1%)	141 (35.1%)	47 (35.1%)
T-category			
T1	113 (21.1%)	87 (21.6%)	26 (19.4%)
T2	219 (40.9%)	156 (38.8%)	63 (47.0%)
T3	116 (21.6%)	91 (22.6%)	25 (18.7%)
T4	86 (16.0%)	67 (16.7%)	19 (14.2%)
Tx	2 (0.4%)	1 (0.2%)	1 (0.7%)
N-category			
N0	19 (3.5%)	14 (3.5%)	5 (3.7%)

Appendix B (continued)

TABLE V: MAIN FEATURE DEMOGRAPHICS OF THE MDACC OPC DATASET FOR EACH TREATMENT JUNCTION.

Characteristic	*All Patients (536)*	*Training set (402)*	*Testing set (134)*
N1	62 (11.6%)	39 (9.7%)	23 (17.2%)
N2	438 (81.7%)	336 (83.6%)	102 (76.1%)
N3	17 (3.2%)	13 (3.2%)	4 (3.0%)
N-category (8th edition)			
N0	20 (3.7%)	14 (3.5%)	6 (4.5%)
N1	249 (46.5%)	181 (45%)	68 (50.7%)
N2	250 (46.6%)	194 (48.3%)	56 (41.8%)
N3	17 (3.2%)	13 (3.2%)	4 (3.0%)
AJCC 7th edition			
II	9 (1.7%)	6 (1.5%)	3 (2.2%)
III	64 (11.9%)	39 (9.7%)	25 (18.7%)
IV	463 (86.4%)	357 (88.8%)	106 (79.1%)
AJCC 8th edition			
I	186 (34.8%)	137 (34.2%)	49 (36.8%)
II	81 (15.2%)	63 (15.7%)	18 (13.5%)
III	64 (12.0%)	44 (11.0%)	20 (15.0%)
IV	203 (38.0%)	157 (39.2%)	46 (34.6%)
N/A	2	1	1
Smoking status at Diagnosis			
Current	115 (21.5%)	85 (21.1%)	30 (22.4%)
Former	203 (37.9%)	151 (37.6%)	52 (38.8%)
Never	218 (40.7%)	166 (41.3%)	52 (38.8%)
Smoking status (Packs/Year)			
Mean (SD)	17.7 (23.7)	16.7 (22.9)	20.5 (26.0)
N/A	28	21	7
Aspiration rate Pre-therapy			
No	520 (97.0%)	388 (96.5%)	132 (98.5%)
Yes	16 (3.0%)	14 (3.5%)	2 (1.5%)
Number of Affected Lymph nodes			

Appendix B (continued)

TABLE V: MAIN FEATURE DEMOGRAPHICS OF THE MDACC OPC DATASET FOR EACH TREATMENT JUNCTION.

Characteristic	*All Patients (536)*	*Training set (402)*	*Testing set (134)*
Mean (SD)	2.0 (1.3)	2.1 (1.3)	1.8 (1.0)
Tumor Laterality			
Bilateral	21 (3.9%)	16 (4.0%)	5 (3.7%)
Left	242 (45.1%)	188 (46.8%)	54 (40.3%)
Right	273 (50.9%)	198 (49.3%)	75 (56.0%)
Tumor subsite			
Base Of Tongue	266 (49.6%)	204 (50.7%)	62 (46.3%)
GlossoPharyngeal Sulcus	10 (1.9%)	10 (2.5%)	-
Not Otherwise Specified	31 (5.8%)	24 (6.0%)	7 (5.2%)
Soft Palate	6 (1.1%)	6 (1.5%)	-
Tonsil	223 (41.6%)	158 (39.3%)	65 (48.5%)
Race			
African American / Black	16 (3.0%)	10 (2.5%)	6 (4.5%)
Asian	4 (0.7%)	3 (0.7%)	1 (0.7%)
Hispanic / Latino	21 (3.9%)	17 (4.2%)	4 (3.0%)
Not Otherwise Specified	5 (0.9%)	3 (0.7%)	2 (1.5%)
Native American	1 (0.2%)	1 (0.2%)	-
White / Caucasian	489 (91.2%)	368 (91.5%)	121 (90.3%)
Post-Induction Therapy variables (after D1 and before D2)			
Prescribed Chemo			
None	342 (63.8%)	250 (62.2%)	92 (68.7%)
Doublet	41 (7.6%)	32 (8.0%)	9 (6.7%)
Triplet	143 (26.7%)	111 (27.6%)	32 (23.9%)
Quadruplet	7 (1.3%)	7 (1.7%)	-
Not Otherwise Specified	3 (0.6%)	2 (0.5%)	1 (0.7%)
Chemo Modification			
Yes	85 (15.9%)	65 (16.2%)	20 (14.9%)
No	451 (84.1%)	337 (83.8%)	114 (85.1%)
Modification Type			

Appendix B (continued)

TABLE V: MAIN FEATURE DEMOGRAPHICS OF THE MDACC OPC DATASET FOR EACH TREATMENT JUNCTION.

Characteristic	*All Patients (536)*	*Training set (402)*	*Testing set (134)*
No Dose Adjustment	451 (84.1%)	336 (83.6%)	115 (85.8%)
Dose Modified	21 (3.9%)	16 (4.0%)	5 (3.7%)
Dose Delayed	10 (1.9%)	9 (2.2%)	1 (0.7%)
Dose Cancelled	18 (3.4%)	13 (3.2%)	5 (3.7%)
Dose Delayed & Modified	6 (1.1%)	5 (1.2%)	1 (0.7%)
Regimen Modification	29 (5.4%)	22 (5.5%)	7 (5.2%)
Unknown	1 (0.2%)	1 (0.2%)	-
Dose Limiting Toxicity			
No	441 (82.3%)	329 (81.8%)	112 (83.6%)
Yes	95 (17.7%)	73 (18.2%)	22 (16.4%)
DLT - Dermatological			
0	506 (94.4%)	383 (95.3%)	123 (91.8%)
1	24 (4.5%)	15 (3.7%)	9 (6.7%)
2	4 (0.7%)	2 (0.5%)	2 (1.5%)
3	2 (0.4%)	2 (0.5%)	-
DLT - Neurological			
0	515 (96.1%)	386 (96.0%)	129 (96.3%)
1	17 (3.2%)	13 (3.2%)	4 (3.0%)
2	3 (0.6%)	2 (0.5%)	1 (0.7%)
3	1 (0.2%)	1 (0.2%)	-
DLT- Gastrointestinal			
0	504 (94.0%)	377 (93.8%)	127 (94.8%)
1	27 (5.0%)	20 (5.0%)	7 (5.2%)
2	2 (0.4%)	2 (0.5%)	-
3	3 (0.6%)	3 (0.7%)	-
DLT - Hematological			
0	509 (95.0%)	383 (95.3%)	126 (94.0%)
1	26 (4.9%)	18 (4.5%)	8 (6.0%)
4	1 (0.2%)	1 (0.2%)	-

Appendix B (continued)

TABLE V: MAIN FEATURE DEMOGRAPHICS OF THE MDACC OPC DATASET FOR EACH TREATMENT JUNCTION.

Characteristic	*All Patients (536)*	*Training set (402)*	*Testing set (134)*
DLT - Nephrological			
0	533 (99.4%)	399 (99.3%)	134 (100.0%)
1	3 (0.6%)	3 (0.7%)	-
DLT - Vascular			
0	534 (99.6%)	401 (99.8%)	133 (99.3%)
1	1 (0.2%)	-	1 (0.7%)
3	1 (0.2%)	1 (0.2%)	-
DLT - Infection (Pneumonia)			
0	535 (99.8%)	401 (99.8%)	134 (100.0%)
1	1 (0.2%)	1 (0.2%)	-
DLT - Grade			
0	446 (83.2%)	334 (83.1%)	112 (83.6%)
1	7 (1.3%)	6 (1.5%)	1 (0.7%)
2	33 (6.2%)	26 (6.5%)	7 (5.2%)
3	41 (7.6%)	29 (7.2%)	12 (9.0%)
4	9 (1.7%)	7 (1.7%)	2 (1.5%)
Imaging			
No	342 (63.8%)	250 (62.2%)	92 (68.7%)
Yes	194 (36.2%)	152 (37.8%)	42 (31.3%)
Complete Response (CR) Primary			
0	452 (84.3%)	335 (83.3%)	117 (87.3%)
1	84 (15.7%)	67 (16.7%)	17 (12.7%)
CR Nodal			
0	520 (97.0%)	388 (96.5%)	132 (98.5%)
1	16 (3.0%)	14 (3.5%)	2 (1.5%)
Parietal Response (PR) Primary			
0	447 (83.4%)	332 (82.6%)	115 (85.8%)
1	89 (16.6%)	70 (17.4%)	19 (14.2%)
PR Nodal			

Appendix B (continued)

TABLE V: MAIN FEATURE DEMOGRAPHICS OF THE MDACC OPC DATASET FOR EACH TREATMENT JUNCTION.

Characteristic	*All Patients (536)*	*Training set (402)*	*Testing set (134)*
0	380 (70.9%)	277 (68.9%)	103 (76.9%)
1	156 (29.1%)	125 (31.1%)	31 (23.1%)
Stable Disease (SD) Primary			
0	525 (97.9%)	394 (98.0%)	131 (97.8%)
1	11 (2.1%)	8 (2.0%)	3 (2.2%)
SD Nodal			
0	526 (98.1%)	396 (98.5%)	130 (97.0%)
1	10 (1.9%)	6 (1.5%)	4 (3.0%)
Post-Concurrent Chemotherapy variables (after D2 and before D3)			
CC Regimen			
None	126 (23.5%)	89 (22.1%)	37 (27.6%)
Platinum Based	257 (47.9%)	198 (49.3%)	59 (44.0%)
Cetuximab Based	129 (24.1%)	95 (23.6%)	34 (25.4%)
Other	24 (4.5%)	20 (5.0%)	4 (3.0%)
CC modification			
No	437 (81.5%)	325 (80.8%)	112 (83.6%)
Yes	99 (18.5%)	77 (19.2%)	22 (16.4%)
CR Primary 2			
0	85 (15.9%)	65 (16.2%)	20 (14.9%)
1	450 (84.1%)	336 (83.8%)	114 (85.1%)
N/A	1	1	-
CR Nodal 2			
0	289 (53.9%)	216 (53.7%)	73 (54.5%)
1	247 (46.1%)	186 (46.3%)	61 (45.5%)
PR Primary 2			
0	459 (85.6%)	344 (85.6%)	115 (85.8%)
1	77 (14.4%)	58 (14.4%)	19 (14.2%)
PR Nodal 2			
0	279 (52.1%)	211 (52.5%)	68 (50.7%)

Appendix B (continued)

TABLE V: MAIN FEATURE DEMOGRAPHICS OF THE MDACC OPC DATASET FOR EACH TREATMENT JUNCTION.

Characteristic	*All Patients (536)*	*Training set (402)*	*Testing set (134)*
1	257 (47.9%)	191 (47.5%)	66 (49.3%)
SD Primary 2			
0	534 (99.6%)	400 (99.5%)	134 (100.0%)
1	2 (0.4%)	2 (0.5%)	-
SD Nodal 2			
0	526 (98.1%)	396 (98.5%)	130 (97.0%)
1	10 (1.9%)	6 (1.5%)	4 (3.0%)
DLT - Dermatological 2			
No	522 (97.4%)	392 (97,5%)	130 (97.0%)
Yes	14 (2.6%)	10 (2.5%)	4 (3.0%)
DLT - Neurological 2			
No	516 (96.3%)	386 (96.0%)	130 (97.0%)
Yes	20 (3.7%)	16 (4.0%)	4 (3.0%)
DLT - Gastrointestinal 2			
No	508 (94.8%)	380 (94.5%)	128 (95.5%)
Yes	28 (5.2%)	22 (5.5%)	6 (4.5%)
DLT - Hematological 2			
No	512 (95.5%)	382 (95.0%)	130 (97.0%)
Yes	24 (4.5%)	20 (5.0%)	4 (3.0%)
DLT - Nephrological 2			
No	529 (98.7%)	397 (98.8%)	132 (98.5%)
Yes	7 (1.3%)	5 (1.2%)	2 (1.5%)
DLT - Vascular 2			
No	535 (99.8%)	401 (99.8%)	134 (100.0%)
Yes	1 (0.2%)	1 (0.2%)	-
DLT - Other 2			
No	522 (97.4%)	390 (97.0%)	132 (98.5%)
Yes	14 (2.6%)	12 (3.0%)	2 (1.5%)
Decisions			

Appendix B (continued)

TABLE V: MAIN FEATURE DEMOGRAPHICS OF THE MDACC OPC DATASET FOR EACH TREATMENT JUNCTION.

Characteristic	*All Patients (536)*	*Training set (402)*	*Testing set (134)*
Decision 1 (Induction Chemo) Y/N			
No	342 (63.8%)	250 (62.2%)	92 (68.7%)
Yes	194 (36.2%)	152 (37.8%)	42 (31.3%)
Decision 2 (CC / RT alone)			
CC	410 (76.5%)	313 (77.9%)	97 (72.4%)
RT alone	126 (23.5%)	89 (22.1%)	37 (27.6%)
Decision 3 Neck Dissection (Y/N)			
No	425 (79.3%)	318 (79.1%)	107 (79.9%)
Yes	111 (20.7%)	84 (20.9%)	27 (20.1%)
Final Outcomes (after D3)			
Overall Survival (4 Years)			
Alive	457 (85.3%)	344 (85.6%)	113 (84.3%)
Dead	79 (14.7%)	58 (14.4%)	21 (15.7%)
Feeding tube 6 months			
No	438 (81.7%)	325 (80.8%)	113 (84.3%)
Yes	98 (18.3%)	77 (19.2%)	21 (15.7%)
Aspiration rate Post-therapy			
No	438 (81.7%)	323 (80.3%)	115 (85.8%)
Yes	98 (18.3%)	79 (19.7%)	19 (14.2%)
Dysphagia			
No	382 (71.3%)	280 (69.7%)	102 (76.1%)
Yes	154 (28.7%)	122 (30.3%)	32 (23.9%)

Appendix C

RADIOMICS FEATURES EXTRACTION

The 3D volumes of interest (VOIs), including the gross primary tumor volumes (GTVp), were manually segmented by a radiation oncologist, and then inspected by a second radiation oncologist within the commercially available contouring software (Velocity AI v3.0.1). The generated VOIs and CT images were exported in the format of DICOM and DICOM-RTSTRUCT to be used for radiomics features extraction. The primary tumor volumes (GTVp) were contoured based on the ICRU 62/83 definition [35]. Radiomics analysis was performed using the freely available open source software ”Imaging Biomarker Explorer” (IBEX), which was developed by the University of Texas MD Anderson Cancer Center and utilizes the Matlab platform (Mathworks Inc, Natick, VA). The CT images in the format of DICOM and the GTVp contours in the format DICOMRTSTRUCT were imported into IBEX. Extracted features represent the intensity, shape, and texture of the primary tumor.

CITED LITERATURE

1. Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P.: Rl^2: Fast reinforcement learning via slow reinforcement learning, 2016.

2. Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M.: Learning to reinforcement learn, 2017.

3. Heess, N., Hunt, J. J., Lillicrap, T. P., and Silver, D.: Memory-based control with recurrent neural networks, 2015.

4. Chakraborty, B. and Murphy, S. A.: Dynamic treatment regimes. Annual Review of Statistics and Its Application, 1(1):447–464, 2014.

5. Schulte, P. J., Tsiatis, A. A., Laber, E. B., and Davidian, M.: Q- and A-learning Methods for Estimating Optimal Dynamic Treatment Regimes. Stat Sci, 29(4):640–661, Nov 2014.

6. Nahum-Shani, I., Qian, M., Almirall, D., Pelham, W., Gnagy, B., Fabiano, G., Waxmonsky, J., Yu, J., and Murphy, S.: Q-learning: A data analysis method for constructing adaptive interventions. Psychological Methods, 17(4):478–494, December 2012. Copyright: Copyright 2014 Elsevier B.V., All rights reserved.

7. Jain, S., Popple, R., Szychowski, J., Sen, B., Locher, J. L., and Kilgore, M. L.: Radiation Oncologist Characteristics and their Association with Outcomes in Patients with Head and Neck Cancer. Pract Radiat Oncol, 9(3):e322–e330, May 2019.

8. Adelstein, D. J., Ridge, J. A., Gillison, M. L., Chaturvedi, A. K., D'Souza, G., Gravitt, P. E., Westra, W., Psyrri, A., Martin Kast, W., Koutsky, L. A., Giuliano, A., Krosnick, S., Trotti, A., Schuller, D. E., Forastiere, A., and Dansky Ullmann, C.: Head and neck squamous cell cancer and the human papillomavirus: Summary of a national cancer institute state of the science meeting, november 9–10, 2008, washington, d.c. Head & Neck, 31(11):1393–1422, 2009.

9. Sheu, T., Vock, D., Mohamed, A., Gross, N., Mulcahy, C., Zafereo, M., Gunn, G., Garden, A., Sevak, P., Phan, J., Lewin, J., Frank, S., Beadle, B., Morrison, W., Lai, S., Hutcheson, K., Marai, G., Canahuate, G., Kies, M., El-Naggar, A., Weber, R.,

Rosenthal, D., and Fuller, C.: Conditional survival analysis of patients with locally advanced laryngeal cancer: Construction of a dynamic risk model and clinical nomogram. Scientific Reports, 7, March 2017.

10. Luciani, T., Wentzel, A., Elgohari, B., Elhalawani, H., Mohamed, A., Canahuate, G., Vock, D., Fuller, C., and Marai, G.: A spatial neighborhood methodology for computing and analyzing lymph node carcinoma similarity in precision medicine. Journal of Biomedical Informatics: X, 5:100067, 2020.

11. Wentzel, A., Luciani, T., van Dijk, L. V., Taku, N., Elgohari, B., Mohamed, A. S. R., Canahuate, G., Fuller, C. D., Vock, D. M., and Marai, G. E.: Precision association of lymphatic disease spread with radiation-associated toxicity in oropharyngeal squamous carcinomas. medRxiv, 2020.

12. Marai, G. E., Ma, C., Burks, A., Pellolio, F., Canahuate, G., Vock, D., Mohamed, A., and Fuller, C.: Precision risk analysis of cancer therapy with interactive nomograms and survival plots. IEEE Transactions on Visualization and Computer Graphics, PP:1–1, 03 2018.

13. Hochreiter, S., Younger, A. S., and Conwell, P. R.: Learning to learn using gradient descent. In IN LECTURE NOTES ON COMP. SCI. 2130, PROC. INTL. CONF. ON ARTI NEURAL NETWORKS (ICANN-2001, pages 87–94. Springer, 2001.

14. Sutton, R. S. and Barto, A. G.: Reinforcement Learning: An Introduction. The MIT Press, second edition, 2018.

15. Daw, N. D., Gershman, S. J., Seymour, B., Dayan, P., and Dolan, R. J.: Model-based influences on humans' choices and striatal prediction errors. Neuron, 69(6):1204–1215, Mar 2011.

16. Moodie, E. E., Chakraborty, B., and Kramer, M. S.: Q-learning for estimating optimal dynamic treatment rules from observational data. Can J Stat, 40(4):629–645, Dec 2012.

17. Barrett, L. and Narayanan, S.: Learning all optimal policies with multiple criteria. In Proceedings of the 25th International Conference on Machine Learning, ICML '08, page 41–47, New York, NY, USA, 2008. Association for Computing Machinery.

CITED LITERATURE (continued)

18. Lizotte, D. J., Bowling, M. H., and Murphy, S. A.: Efficient reinforcement learning with multiple reward functions for randomized controlled trial analysis. In ICML, pages 695–702, 2010.

19. LizotteDaniel, J., BowlingMichael, and MurphySusan, A.: Linear fitted-q iteration with multiple reward functions. Journal of Machine Learning Research, 2012.

20. Tosado, J., Zdilar, L., Elhalawani, H., Elgohari, B., Vock, D. M., Marai, G. E., Fuller, C., Mohamed, A. S. R., and Canahuate, G.: Clustering of Largely Right-Censored Oropharyngeal Head and Neck Cancer Patients for Discriminative Groupings to Improve Outcome Prediction. Sci Rep, 10(1):3811, 03 2020.

21. Wong, A. J., Kanwar, A., Mohamed, A. S., and Fuller, C. D.: Radiomics in head and neck cancer: from exploration to application. Transl Cancer Res, 5(4):371–382, Aug 2016.

22. Bogowicz, M., Leijenaar, R. T. H., Tanadini-Lang, S., Riesterer, O., Pruschy, M., Studer, G., Unkelbach, J., Guckenberger, M., Konukoglu, E., and Lambin, P.: Post-radiochemotherapy PET radiomics in head and neck cancer - The influence of radiomics implementation on the reproducibility of local control tumor models. Radiother Oncol, 125(3):385–391, 12 2017.

23. Vallières, M., Kay-Rivest, E., Perrin, L. J., Liem, X., Furstoss, C., Aerts, H. J. W. L., Khaouam, N., Nguyen-Tan, P. F., Wang, C. S., Sultanem, K., Seuntjens, J., and El Naqa, I.: Radiomics strategies for risk assessment of tumour failure in head-and-neck cancer. Sci Rep, 7(1):10117, 08 2017.

24. Parmar, C., Leijenaar, R. T., Grossmann, P., Rios Velazquez, E., Bussink, J., Rietveld, D., Rietbergen, M. M., Haibe-Kains, B., Lambin, P., and Aerts, H. J.: Radiomic feature clusters and prognostic signatures specific for Lung and Head Neck cancer. Sci Rep, 5:11044, Jun 2015.

25. Parmar, C., Grossmann, P., Rietveld, D., Rietbergen, M. M., Lambin, P., and Aerts, H. J.: Radiomic Machine-Learning Classifiers for Prognostic Biomarkers of Head and Neck Cancer. Front Oncol, 5:272, 2015.

26. Jethanandani, A., Lin, T. A., Volpe, S., Elhalawani, H., Mohamed, A. S. R., Yang, P., and Fuller, C. D.: Exploring Applications of Radiomics in Magnetic Resonance Imaging of Head and Neck Cancer: A Systematic Review. Front Oncol, 8:131, 2018.

27. Elhalawani, H., Kanwar, A., Mohamed, A. S. R., White, A., Zafereo, J., Wong, A., Berends, J., Abohashem, S., Williams, B., Aymard, J. M., Perni, S., Messer, J., Warren, B., Youssef, B., Yang, P., Meheissen, M. A. M., Kamal, M., Elgohari, B., Ger, R. B., Cardenas, C. E., Fave, X., Zhang, L., Mackin, D., Marai, G. E., Vock, D. M., Canahuate, G. M., Lai, S. Y., Gunn, G. B., Garden, A. S., Rosenthal, D. I., Court, L., and Fuller, C. D.: Investigation of radiomic signatures for local recurrence using primary tumor texture analysis in oropharyngeal head and neck cancer patients. Sci Rep, 8(1):1524, 01 2018.

28. van Dijk, L. V., Thor, M., Steenbakkers, R. J. H. M., Apte, A., Zhai, T. T., Borra, R., Noordzij, W., Estilo, C., Lee, N., Langendijk, J. A., Deasy, J. O., and Sijtsema, N. M.: Parotid gland fat related Magnetic Resonance image biomarkers improve prediction of late radiation-induced xerostomia. Radiother Oncol, 128(3):459–466, 09 2018.

29. van Dijk, L. V., Langendijk, J. A., Zhai, T. T., Vedelaar, T. A., Noordzij, W., Steenbakkers, R. J. H. M., and Sijtsema, N. M.: Delta-radiomics features during radiotherapy improve the prediction of late xerostomia. Sci Rep, 9(1):12483, 08 2019.

30. Platt, J. C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In ADVANCES IN LARGE MARGIN CLASSIFIERS, pages 61–74. MIT Press, 1999.

31. Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., and Kavukcuoglu, K.: Asynchronous methods for deep reinforcement learning, 2016.

32. Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A. J., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., Kumaran, D., and Hadsell, R.: Learning to navigate in complex environments, 2017.

33. Hochreiter, S. and Schmidhuber, J.: Long short-term memory. Neural computation, 9:1735–80, 12 1997.

34. Kingma, D. P. and Ba, J.: Adam: A method for stochastic optimization, 2017.

35. authors listed, N.: 4. Definition of volumes. J ICRU, 10(1):41–53, Apr 2010.

VITA

NAME	Elisa Tardini
EDUCATION	
	Master of Science in "Computer Science", University of Illinois at Chicago, May 2021, USA
	Specialization Degree in "Computer Science and Engineering ", Jul 2021, Polytechnic of Milan, Italy
	Bachelor's Degree in "Computer Science and Engineering", Jul 2018, Polytechnic of Milan, Italy
LANGUAGE SKILLS	
Italian	Native speaker
English	Full working proficiency
	2018 - IELTS examination (8.0/9)
	2015 - Cambridge Certificate of Proficiency in English (A)
	2014 - Cambridge Certificate of Advanced English (A)
SCHOLARSHIPS	
Spring 2020-Spring 2021	Research Assistantship (RA) position (20 hours/week) with full tuition waiver plus monthly stipend
Fall 2019	Italian scholarship for TOP-UIC students
Fall 2016	Award for top freshmen at Polytechnic of Milan
TECHNICAL SKILLS	
Basic level	Computer architectures, computer security, distributed systems, CUDA, Matlab
Average level	Databases and SQL, C, Java
Advance level	Data Science and Machine Learning, Causal Inference, Python
WORK EXPERIENCE AND PROJECTS	
Jan 2020 - present	Reinforcement Learning for Dynamic Treatment Regimes

VITA (continued)

	Application of Supervised and Reinforcement Learning algorithms to a dataset of Oropharyngeal Squamous cell Carcinoma patients treated at the MD Anderson Cancer Center. Development and evaluation of a treatment simulator and custom RL algorithms for the personalization of treatment regimes.
Aug 2019 - Dec 2019	Evaluating Human Performance Using Heterogeneous Treatment Effects
	Development and evaluation of a Supervised Learning model for the assessment of Heterogeneous Treatment Effects in a dataset of Stack-exchange votes, questions, answers and acceptances. Used model to analyse effects of question acceptance on voting and of question voting on acceptance.
April 2019 - May 2019	Road traffic forecasting
	Design and evaluation of a time series Supervised Learning model for predicting road traffic speeds and traffic based on road events and weather conditions.