Towards Hybrid-AI in Imaging using VoxLogicA

Gina Belmonte[1], Laura Bussi[2], Vincenzo Ciancia[2], Diego Latella[⋆2], and Mieke Massink[2]

[1] Azienda USL Toscana Nordovest, Ospedale San Luca, Lucca, Italy
gina.belmonte@uslnordovest.toscana.it
[2] Consiglio Nazionale delle Ricerche, Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", Pisa, Italy
{Laura.Bussi, Vincenzo.Ciancia, Diego.Latella, Mieke.Massink}@cnr.it

Abstract. We present the design of a meta-programming system for hybrid AI, integrating spatial model checking and machine learning. The proposed system architecture blends together different programming languages and execution technologies using a simplified, declarative meta-language. The design features a global-model-checking-alike execution model, backed up by a microservices architecture. The system is meant to be a follow up to the spatial model checker VoxLogicA currently used for research on declarative medical image analysis, aimed at *explainable by construction* artificial intelligence.

Keywords: Spatial logic · Spatial Model Checking · Machine Learning · Hybrid-AI · Medical Imaging

1 Introduction

This work stems from a simple consideration: in the age of Artificial Intelligence, the procedures which we depend upon should be *accountable*, in a way that humans can understand, study, debate, defend, and improve them.

Clearly, not all aspects of a procedure can be *explainable* in a formal way. Some features of reasoning are very difficult, if not impossible, to formalise. However, reasoning should be *as transparent as possible* and, in the cases where explainable rules are present, these *must* be taken into account. Rules can be used at least for three major purposes: to instruct training of the machines, for instance, by *augmenting* the *ground truth* with the output of pre-computed rules; to *monitor* the results by verifying to which extent known *protocols* and *guidelines* are respected, and to *compose* pipelines consisting of several independent machine learning modules, in a way that makes sense, intuitively, and breaks down a complex problem into clearly defined smaller tasks.

Such activities have been *formally structured* way before the introduction of modern Artificial Intelligence. Human intelligence alone can be very efficient at performing a variety of tasks. But measurements, indicators, and, broadly

⋆ Diego Latella has contributed to the research presented in this paper when he was still Senior Researcher with CNR. Since Sep. 1, 2024 he has retired.

speaking, technological aids, are typically used to inform human activities. This is akin to the process of dataset augmentation: both are meant to provide artificially, mechanically derived features that are not naturally present in the input data, in order to facilitate understanding of the structure of inputs. It is also true that, when coordinating human activities, protocols are often formalised (think for instance of *business process modelling* or *clinical protocols*), so that smaller independent tasks concur to the realisation of the final objective. Finally, *quality assurance* verifies the outcome of a human activity against known metrics. It is quite reasonable to expect that similarly, also Artificial Intelligence should make use of technological aids to augment its inputs, to break down tasks, and to verify the quality of the results against known indicators.

In Formal Methods, planning, monitoring, and measuring, are frequently performed using *declarative* programming paradigms, employing concise, unambiguous, mathematically specified languages such as *modal logics* or *process calculi*. In particular, *model checking* enables fully automated verification of properties specified using modal logics.

In this work, we maintain that deep learning training and prediction can, and should, be *hybridised* with declarative programming and model checking, in order to achieve the three stated goals of *declarative data augmentation*, *declarative monitoring of machine learning predictions*, and *declarative composition of machine learning models*.

The obtained methodology is *explainable by construction*, and it makes use of declarative programming as much as possible, even if it does not strictly require interpretability of all components of a pipeline, staying open towards state-of-the-art models that do not (yet?) possess this feature, but can be incorporated in a hybrid specification to monitor and assess the quality of their results.

Image analysis provides a near-to-perfect setting for experimentation in this research line, as in this application domain, classical declarative programming paradigms, and black-box approaches, can happily coexist. On the one hand, many features of an image (think e.g. of: "lesions in a magnetic resonance scan of a human brain") are very well recognised using machine learning. For the purpose of our work, such type of tasks could be handled by a fully automated pipeline such as `nnU-Net` [29,30], which does not require manual tuning of hyperparameters in the training phase, and is widely recognised as the state-of-the-art for medical image segmentation. On the other hand, there are purely formal aspects of an image that *cannot* be specified using `nnU-Net` alone, without specialising the training process, and creating additional ground truth which is not available. For instance, consider the task of identifying, in a large dataset of photos of human faces, images with a specified "defect", such as those showing only one eye, or people wearing glasses, or with / without hair, and so on. The formulation of such problems combine black-box concepts (such as "eye" or "glasses") and declarative specification (the number of eyes, the presence or absence of a feature, the colour of hair, etc.).

So how are we going to implement a system that permits arbitrary combinations of training, prediction, and declarative rules? In recent work, starting

from [17] and the extended version [18], together with several coauthors, we investigated a novel variant of the model checking problem moving from checking temporal logic properties to *spatial* logic properties (and, in fact, also to the combination of reasoning on time and space in spatio-temporal model checking [15,20,16,21]). The proposed spatial logic for image analysis, SLCS, encompasses several domain oriented operators to accommodate the level of abstract spatial reasoning by domain experts [7,2,5]. The combination of basic logical operators, spatial operators and domain oriented operators provides powerful building blocks to develop concise, human readable and explainable image segmentation methods. In the spatial model checker `VoxLogicA` (see [19]), efficient execution is backed by the implementation of most logical primitives via state-of-the-art imaging libraries, and more recently, by using the GPU [11].

In this work, we propose the design of a novel system devoted to Hybrid Artificial Intelligence (HAI) for image analysis, encompassing the technology which lies at the grounds of `VoxLogicA`[3], merging it with two new language primitives that correspond to *training* and *prediction* of a black-box segmentation module, implemented using `nnU-Net`.

Such design is prominently based on a *microservices* architecture [23]. This choice is driven by the necessity of decoupling very different run-time architectures (for instance, the training and prediction modules that are implemented in `python`, and the on-GPU primitives, whose natural implementation language is `C++`), but also since independent deployment paves the way to a *privacy-first* and *local-first* approach, where datasets are "as open as possible and as closed as necessary" and therefore, modules that have access to full datasets, such as the training nodes, ought to be placed close to the data, which should not be transferred to sites which are only elaborating anonymised results and statistics.

We emphasize that, to the best of our knowledge, current image analysis systems are not based upon declarative programming, but rather on classical programming paradigms. On the one hand, traditional programming (e.g., using the `python` programming language), is a difficult skill to master. In contrast, the approach we propose is aimed at simple specifications that are easy to interpret, share, maintain and improve. On the other hand, just like in many other fields of Computer Science (consider e.g. relational database systems or query languages for semi-structured data), a declarative approach is amenable to fully automated optimization (e.g. parallel execution) and is therefore amenable of performance improvements that are difficult or impossible to obtain in more classical ways.

We envision that such fully-automated hybrid AI paradigm will be used to perform, among other tasks, image segmentation, ground truth augmentation, monitoring, and quality assurance, and will enhance accountability. This will be achieved by providing users with a form of explainability of the results which is "permissive", in that it does not force an impossible-to-grasp intuition on the inner working of non-rule-based features, but on the other hand, it permits to coordinate such features in unambiguous, human readable, automatically exe-

[3] In particular, the model checking core featuring memoization and parallel execution of imaging primitives, and the GPU-accelerated implementation of some of them.

cutable specification language which can be as transparent as the whole process *can* be.

Related work. In [17,18] the Spatial Logic for Closure Spaces (SLCS), and related efficient model checking algorithms, have been proposed that use *closure spaces* [24,25,34,26], a generalisation of topological spaces, as the underlying model. In [16] a temporal extension of spatial model checking was introduced, combining Computation Tree Logic with the spatial operators of SLCS. Spatio-temporal model checking has recently been applied in a variety of domains, ranging from Collective Adaptive Systems and the Internet-of-Things [15,20,21,38,37] to signals [32]. Spatial model checking has been used for the analysis of medical images [2,3,4,6,7].

In [2] a first feasibility study of the application of spatial model checking on the segmentation of brain lesions is described. That study has been performed with a predecessor of `VoxLogicA` for SLCS, namely the general purpose spatio-temporal model checker `topochecker`. The analysis concerned several 2D and 3D MR images of patients with high-grade glioma, a serious form of brain cancer. In [7] the work on spatial model-checking has been taken further with the development of the domain specific, and much more efficient, *spatial* model checker `VoxLogicA` and its application on the BraTS 2017 dataset [36], a dataset for brain tumour segmentation challenges. The obtained results are in line with best-in-class machine learning methods, and human manual segmentation. In [6] a feasibility study of the segmentation of tissues present in the *normal* brain was undertaken, restricted to the analysis of the first 2 of the 20 patients provided by the BrainWeb dataset[4].

2 Background

Spatial Logic is a classical branch of Modal Logic [1] and at the same time, has been a leading research topic in AI and Robotics [22]. Spatial model checking [17,18,2,7], instead, is a novel variant of model checking in which local image features (intensity, colour, texture) and spatial/topological characteristics (relative distance, contact, connectedness), expressed as formulas of a spatial logic, can be checked automatically for a spatial model. Typical spatial models may be represented as graphs. Digital images can be seen as particular forms of regular graphs, or grids, where each node in the graph represents a pixel or voxel. Pixels or voxels can have particular features (e.g. their luminosity or intensity or their colour). These basic features can be used in the logical properties.

Essentially, a spatial model checker takes as input a digital image and a logic property (for example, a query involving some aspects of the pixels, such as their intensity, and aspects of those in their direct neighbourhood), and it gives as an output a resulting image where all the pixels of the input image that satisfy the property are shown in a user-specified way, e.g. in a certain colour. The resulting

[4] See http://www.bic.mni.mcgill.ca/brainweb/.

images are meant to be visualised as (semi-transparent) coloured layers on top of the original images.

The recently developed spatial model checker VoxLogicA[5] together with its specification language ImgQL ("image query language"), a customisation of SLCS for medical images, has been optimised for operating on such images, e.g. MRI or Computer Tomography (CT) scans (see [19] for a tutorial). Common similarity indexes, such as Dice-Sorensen, can be directly defined and used in ImgQL; both the resulting layers and the indexes of interest can be saved on the local file system where the user runs the analysis (see e.g. Alg. 1 lines 16-17).

Algorithm 1: ImgQL specification of a tumour segmentation method.

```
1  // Normalisation through percentiles
2  let pflair = percentiles(flair,brain,0)
3  // High and low thresholds
4  let hI = pflair >. 0.95
5  let vI = pflair >. 0.86
6  // Remove noise (holes) in hI and vI
7  let hyperIntense = flt(5.0,hI)
8  let veryIntense = flt(2.0,vI)
9
10 // Region growing procedure
11 let growTum = grow(hyperIntense, veryIntense)
12 // Statistical texture similarity
13 let tumSim = similarFLAIRTo(growTum)
14 let tumStatCC = smoothen(2.0,tumSim >. 0.6)
15 // Final tumour segmentation
16 let gtv = grow(growTum,tumStatCC)
17 save "gtv.nii.gz" gtv
18 print "00_dice_gtv" diceM(gtv)
```

Figure 1 shows some of the intermediate phases of the procedure presented in Algorithm 1. The obtained results are well in line with the state of the art in segmentation of high-grade brain tumours (glioma), both in terms of accuracy of the segmentation and in terms of computational efficiency, reaching on average an 85% coincidence between the gold standard and the segmentation procedure shown in Figure 1, for the gross tumour volume (and 91% on average for the clinical tumour volume) [7]. Note that these results have been reached using the same ImgQL specification on *all* the relevant cases in the BraTS 2017 dataset (193 cases in total). This was partially made possible also due to the simple, but effective, normalisation procedure using percentiles, shown in the first lines of the ImgQL specification. Moreover, the method supports explainability, easy replicability and exchange of analysis methods between professionals.

[5] VoxLogicA is available at https://github.com/vincenzoml/VoxLogicA

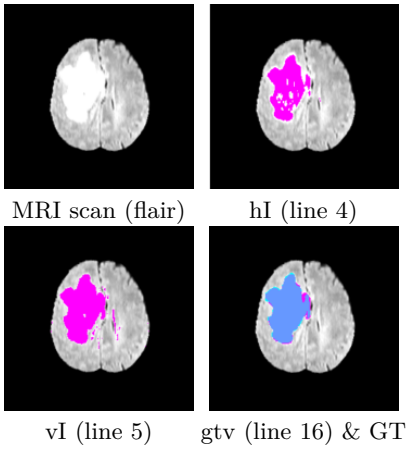

Fig. 1. Resulting example segmentation of a slice of the axial plane of the 3D image `Brats17_2013_7_1`: Overlays `hI` (hyper intense voxels), `vI` (very intense voxels) and `gtv` (gross tumour volume, in pink) and ground truth (GT) (in blue).

The segmentation procedure is concise, explainable and user-centric because its high-level steps fit the abstraction level at which clinicians are assessing an image. One of the phases in the method is based on the region-growing technique [27]. This is a well-established segmentation technique that is still further improved and refined (see for instance [28]).

3 A Microservice Architecture for Hybrid AI

The *Microservices architecture* [23] is an architectural pattern where large applications are not implemented in a monolithic way, but rather as collections of isolated services. This choice aims at good performance, resilience and scalability properties. The distinguishing features of a microservice architecture are:

- each service is developed, deployed and maintained independently;
- services are *loosely coupled*, that is, changes to one service do not propagate, or impose minimal changes, to the others;
- services communicate via lightweight protocols.

3.1 Microservices: Motivations

The current stable version of `VoxLogicA` is a monolithic application. It entirely runs on CPU, exploiting *memoization* and parallel execution on CPU cores, and relies on the state-of-the-art imaging library SimpleITK [39]. The core model checking code is written in `F#`. The application basically provides a single service, which is a global model checker for the language `ImgQL`, that is, the spatial logic SLCS enriched with imaging primitives, thresholds, texture analysis, and

other non-logical operators. Such a core is extensible, and therefore the parallel, memoizing engine has been reused in the experimental on-GPU variant of [11] and in the model checker for polyhedral structures (in particular, 3D meshes) of [8]. The lesson learned, especially from the on-GPU implementation, is that a monolithic architecture is not extensible enough to accommodate the needs of efficient, fully automated, declarative analysis.

To understand the issues we encountered, just a little bit of technical detail will suffice. For instance, the implementation of CPU and GPU primitives in [11] are tightly coupled, and ultimately, in order to implement *garbage collection* of GPU memory, and reuse a memory block in a subsequent operation, it is required to call a *lock* primitive on the central data-structure of the on-GPU implementation, and a *wait* instruction on the GPU in the critical section, at the same time, which is both a performance bottleneck and a source of subtle bugs which are very difficult to analyse.

In the design we propose, `VoxLogicA` becomes a collection of services, communicating via unix sockets or HTTP using a simple `json` [9] description of the portion of the computation that has to be executed on the specific service. Indeed, such an architecture cannot be as efficient as the monolithic core, which communicates over shared memory in different threads of the same system process, when operating on a single image, since the communication costs dominate the processing. However, most of the applications of `VoxLogicA` operate on a *dataset* consisting of hundreds or thousands of images, which are processed independently (e.g. for segmentation purposes) or anyway in a pipeline (for instance to analyse a video, frame by frame, as done in [12]). In that case, each service, when executing a logical primitive, can operate on streams of data, with size proportional to that of the dataset. Such streams are either the dataset itself, or the result of previous primitives which in turn operate on the full dataset. In other words, services are expected to operate on whole datasets and not single items (possibly processing them in chunks due to limited computational resources). This feature is intended to be combined with a per-site caching strategy, ensuring minimization of data transfers. In this setting, the overhead due to the use of inter-process or inter-machine communication is expected to be negligible, with respect to the time needed to process the payload.

3.2 System design

In this section we discuss the proposed system design (see Figure 2). The public interface of the system is represented by the `controller`. The `controller` takes as an input a declarative specification of an image analysis procedure, which also references a dataset of images to be analysed. The model checking engine of the controller transforms the specification into a directed acyclic graph (DAG) of tasks and dependencies (just like it is currently done in `VoxLogicA`). The controller is also responsible for splitting the DAG into separated, connected *fragments*, according to the service(s) that provide the primitives present in each component. If certain logical primitives are implemented by more than one service, the splitting phase can be informed by non-functional properties such

as availability, efficiency, computation cost, and last but not least, privacy and intellectual property matters that may require sensitive primitives (for instance, training) to be executed in the same sites where the data resides. Each fragment is then sent to one of the services, according to the plan. Each service is thus enabled to prepare a computation pipeline for each fragment, which can be reused to process each element of the input data (be it the dataset itself, or the result of other language primitives).

As shown in Figure 2, intermediate results are not sent directly to other services, but rather to a `data manager`, that takes care to store them in a shared database. The data manager is responsible for monitoring data protection policies (also coming from IP and privacy agreements and regulations), and for returning final results to the controller. By using a data manager, the failure of an arbitrary number of services, or runtime errors in the execution of a language primitive, still allow intermediate results, that are independent from the failure, to be computed and returned to the user.

Some further discussion is due on the data storage service. The implementation is intended to be *local-first* [31], that is, the system should run on locally stored data *independently* from availability of the data in a wider network or in cloud storage, but in a way that is *open* to such possibility and can eventually take advantage of it. This is in order to comply with regulations that may even forbid a computing machine to be connected to the internet, if such a machine operates on clinical data, and at the same time, to permit collaborative development among different, federates sites, when this is possible. Indeed, such design aims at a *privacy by design* [13] and *privacy by default*[6] approach, where implementation of our system is directly responsible for making sure that language primitives that may be harmful to the privacy of individuals can only be executed according to regulations. In the first version of the proposed design, such analysis is implemented at system deployment time. Each microservice is replicated and deployed to different locations, both to exploit, for example, multiple cores or multiple GPUs to perform several operations in parallel, and for privacy and IP concerns. In fact, data transfer from a location to another must be authorised by the data manager, which is responsible for guaranteeing the privacy-by-default approach at *run-time*. Investigation on static analysis of the specification language, to provide guarantees directly from the controller and signal inconsistencies and violations at *compile-time* are left for future work.

3.3 Four Fundamental Services

The architecture we propose is meant to be extensible, so that adding a new set of language primitives is just a matter of implementing the primitives themselves inside a microservice, in a programming language of choice, and registering the provided operation names, arity and types within the system so that user requests involving the new primitives can be fulfilled. Therefore, we envisage that,

[6] See EuropeanUnion.(2016)."GeneralDataProtectionRegulation(GDPR)-Regulation(EU)2016/679."OfficialJournaloftheEuropeanUnion.

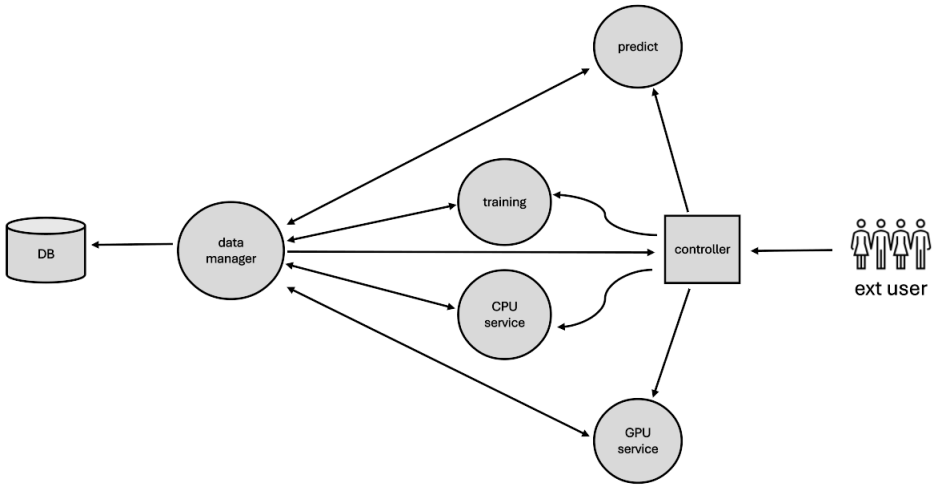

Fig. 2. The microservices architecture of VoxLogicA. In this example instance, there are 4 services ("predict", "training", "CPU imaging", and "GPU imaging"), that implement language primitives; indeed, there could be more. The controller is the public interface of VoxLogicA, and takes care of splitting operations and distribute them over different services. The data manager takes care of storing intermediate and final results, and returning them to the user. The data store is intended to be privacy-aware and local-first. Note that this diagram intentionally *does not* illustrate the spatial distribution and replication of services, that are needed both to accommodate boundaries due to privacy and IPR regulations, and to guarantee availability.

alongside deployment of the system, a plethora of services dedicated to different activities could be deployed. Think, for example, of data-gathering services connected to a PACS ("Picture Archiving and Communication System") [14], or of generative-AI primitives that can create artificial ground truth, for example by fine-tuning and invoking *diffusion models* [35]. However, four services are already quite prominent in the envisaged architecture, and can be described in more detail already in this work.

The `training` service. The `training` service is implemented in `Python`, as this is by far the most used language for machine learning, and the one in which `nnU-Net` is implemented. The training service is separated from prediction, because they share different features. Training primitives are *long-running*, as an invocation can last several days. Therefore, training requires batch execution, and saves intermediate data to disk to avoid restarting from scratch in case of failure. Furthermore, by its own nature, training must access entire datasets, therefore in a privacy-by-default scenario, it is preferably run in the same site where the training data resides. Indeed, since dataset augmentation by means of logical specifications is one of our stated goals, the need to move the training

service close to the data also implies that logic-based modules must be replicated to such sites. The output of training are the *weights* of a deep-learning based module.

The `predict` *service.* The `predict` service is also implemented in `Python`, for basically the same reasons as the `training` service. Prediction is a slow procedure, but still online/interactive. This service needs to use the *weights* of the chosen network, which usually come from a training primitive. Predict alone can be quite precise, but for accountability reasons, we advocate combining predict with logic-based monitoring.

The `on-GPU primitives` *service.* The `on-GPU primitives` encompasses logical operators as described in [11], and some imaging functionalities that can be easily parallelised (e.g. thresholding). The service is designed to be very fast (usually real-time). The on-GPU analysis includes a limited set of primitives, designed to exploit massive parallelisation on GPU, and it is particularly effective on large datasets, where the cost of setting up a computing pipeline and loading data does not dominate the computation time. This service is implemented using low-level data-parallel programming languages inspired by C[7].

The `on-CPU primitives` *service.* On-CPU primitives are slower than those running on GPU, but the range of available operators is much wider, and it includes the use of large state-of-the-art imaging libraries such as SimpleITK, and the possibility of using statistical methods e.g. for texture analysis (see [2]). Also the choice of implementation languages is much less constrained.

By implementing these four services, we envisage the possibility of building efficient, scalable hybrid-AI pipelines that exploit the available computing resources to their best. In the next section we will discuss an example of this kind.

4 Example

In Algorithm 2, we show the specification of an example application scenario, which can be considered an extension of [7] in the direction of hybrid AI (see Algorithm 1 in Section 2). The specification uses the syntax of `VoxLogicA`, but it also assumes that the input to imaging and logical primitives is a whole *dataset*, not a single image.

The idea is that, instead of just using thresholds, two neural networks are used to learn the concepts of *oedema* and *tumour*, separately, also exploiting data augmentation, and then combined using logics as before. The results are then compared to the standard method that just learns the gross tumour volume (the combination of tumour and oedema) in a single training step, without data augmentation.

[7] Such as OpenCL (see https://www.khronos.org/opencl/) and CUDA (see https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html).

Algorithm 2: Brain tumour segmentation using hybrid-AI primitives.

```
1  //data augmentation
2  let augment (dataset,thresholdValue) =
3      merge (dataset,threshold (thresholdValue, dataset))
4  let tumourCoreDatasetAugmented =
5      augment (tumourCoreTrainingDataset,tumourThreshold)
6  let oedemaDatasetAugmented =
7      augment (oedemaTrainingDataset,oedemaThreshold)
8
9  //training
10 let tumourCoreModel = train (tumourCoreDatasetAugmented)
11 let oedemaModel = train (oedemaDatasetAugmented)
12 let nnGtvModel = train (originalTrainingDataset)
13
14 //prediction
15 let inputTumourAugmented =
16     augment (inputDataset,tumourThreshold)
17 let inputOedemaAugmented =
18     augment (inputDataset,oedemaThreshold)
19 let mlTumourCore =
20     predict (tumourCoreModel, inputTumourAugmented)
21 let mlOedema =
22     predict (oedemaModel, inputOedemaAugmented)
23 let nnGtv = predict (nnGtvModel, inputDataset)
24
25 //Logic-based pipeline, results and comparison
26 let oedema = touch (mlOedema, mlTumourCore)
27 let gtv = union(mlTumourCore,oedema)
28 let overlap = diceScore (gtv,nnGtv)
```

Dataset augmentation and training are performed in lines 1-12. The "augment" function is defined in lines 1-2, in order to enrich the dataset with additional input for each case[8] (the `merge` function is meant to take two datasets of the same length, and merge them case by case). The original dataset includes the ground truth relative to the tumour and oedema, respectively. The augmentation is run twice, with two different thresholds for each of the two training processes. Note that the images provided for augmentation purposes correspond to `hyperIntense` and `veryIntense` of Algorithm 1, except that, for simplicity, we have omitted the filtering step, which can as well be added if needed. Augmenting the dataset aims at conditioning the training process so that the

[8] Frequently, dataset augmentation is aimed at adding cases to the dataset (e.g., rotations, rescaling, deformations of existing data points). This can certainly be done with the envisaged architecture, but in this particular example, instead, the dataset is augmented to add derived information to each case. It is noteworty that `nnU-Net` can be trained on an arbitrary number of input images per case in a *multimodal* way.

concept of "hyperIntense" and "veryIntense" are used in predictions based on the model (also the predict primitive will need augmented inputs). Note that this process does not *force* the output of prediction to be *exactly* the hyperintense or veryintense areas of Algorithm 1, but rather, the details are left to the machine learning core. Also note that three models are trained, two using dataset augmentation (`tumourCoreModel` and `oedemaModel`) and one just using the original dataset to learn the gross tumour volume (`nnGtvModel`).

After training, using the resulting model weights, the `predict` primitive is employed in lines 15-23. Then, the tumour and oedema as predicted by the neural networks are combined using the `touch` primitive as in Algorithm 1 (done in line 26). The *gross tumour volume* is then computed in line 27 as the union of `tumour` and `oedema`, and compared with the gross tumour volume as obtained directly from the neural network, by calculating the Dice-Sorensen index of the two images, in line 28. The results in `nnGTV`, `gtv`, and the numeric value of `overlap` are intended to be returned to the user.

In Figure 3, a DAG is shown, corresponding to the specification of Algorithm 2. The DAG has actually been generated using a freshly implemented prototype of the controller depicted in Figure 2, although colours representing the service to which each task is assigned, and the output nodes, have been manually inserted for clarity. Note that the DAG is actually a semantic interpretation of the specification. For instance, `let` bindings and in particular the function `augment` are completely eliminated by the compiler.

5 Discussion

We have presented the design of a first-of-its kind instrument for fully automated image analysis based on a hybrid-AI approach, where topology-based spatial-logical primitives coexist with machine-learning based training and prediction.

Indeed, a similar approach could be pursued by programming directly in a traditional programming language such as python. However, our proposal is that of a *meta-language*, whose execution semantics is distributed across several different microservices. The aim is to *empower* domain experts who are not programmers, to use the heterogeneous landscape of Artificial-Intelligence-based tools and methods available for Image Analysis. Furthermore, fully automated parallelization and the GPU-based acceleration are points in favour of adopting a more high-level approach, with respect to manual programming of such hard and error-prone tasks.

Validation of the proposed methodology will be based on the current brain-imaging related case studies, and on further case studies such as identification of lung diseases.

In future work, also Human-Computer interaction issues should be tackled. In [10] a study has been conducted on the design of a graphical user interface (GUI) prototype that supports the analysis procedure using `VoxLogicA` with minimal impact on the focus and the memory load of domain experts. Such work should be complemented to provide to the user also an overview of the

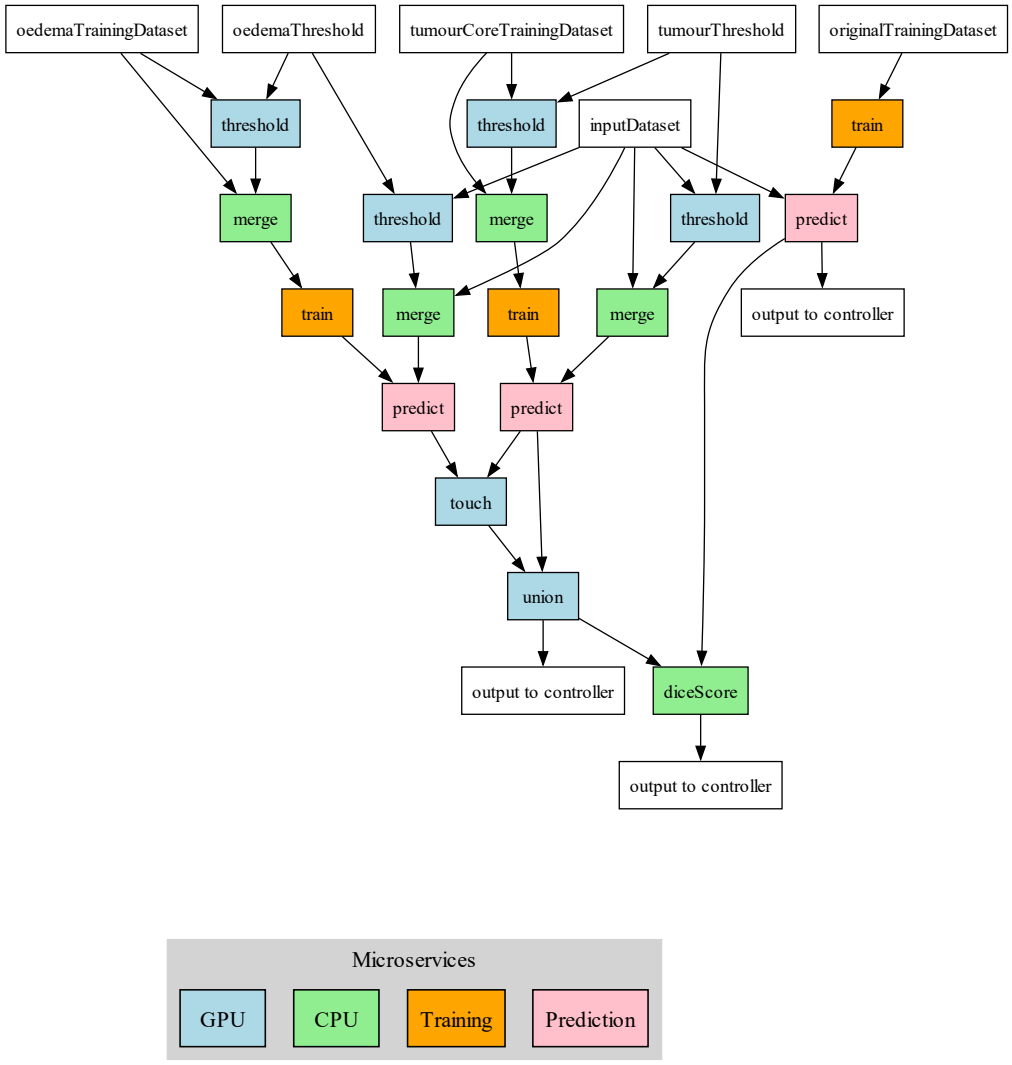

Fig. 3. Directed acyclic graph obtained from Algorithm 2 using a prototype implementation of the new tool. Cyan nodes run on the `GPU` service, orange nodes on the `train` service, pink nodes on the `predict` service and green nodes on the `CPU` service. White nodes represent inputs and outputs.

data distribution across the services, in order to gather a visual feedback about IP and privacy-related errors.

The scope of this work shares motivations with the approach of *Intersymbolic Artificial Intelligence* proposed by A. Platzer in [33]. Therein, a combined symbolic / subsymbolic approach is advocated to encompass both logic-based, exact reasoning and approximate, training-based analysis, by combining building blocks of symbolic programs and subsymbolic algorithms / networks. Indeed, the declarative language of the tool we propose is based on a very similar idea, pairing logical primitives with training and prediction based on subsymbolic methods, and could also be considered an *intersymbolic programming language*.

Summing up, we envisage that a tool for hybrid-AI based imaging may be a valuable addition for the research community, with its unique added-value to be able to encode domain expertise in logic-based form, for the analysis of small and large datasets, and that future research in this line will lead to a fully-distributed, privacy-by-default toolchain.

Acknowledgments. Research partially supported by Bilateral project between CNR (Italy) and SRNSFG (Georgia) "Model Checking for Polyhedral Logic" (#CNR-22-010); European Union - Next GenerationEU - National Recovery and Resilience Plan (NRRP), Investment 1.5 Ecosystems of Innovation, Project "Tuscany Health Ecosystem" (THE), CUP: B83C22003930001; European Union - Next-GenerationEU - National Recovery and Resilience Plan (NRRP) – MISSION 4 COMPONENT 2, INVESTMENT N. 1.1, CALL PRIN 2022 D.D. 104 02-02-2022 – (Stendhal) CUP N. B53D23012850006; MUR project PRIN 2020TL3X8X "T-LADIES"; CNR project "Formal Methods in Software Engineering 2.0", CUP B53C24000720005; Shota Rustaveli National Science Foundation of Georgia grant.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Aiello, M., Pratt-Hartmann, I., Benthem, van, J.: Handbook of Spatial Logics. Springer (2007). https://doi.org/10.1007/978-1-4020-5587-4
2. Banci Buonamici, F., Belmonte, G., Ciancia, V., Latella, D., Massink, M.: Spatial logics and model checking for medical imaging. Int. J. Softw. Tools Technol. Transf. **22**(2), 195–217 (2020), https://doi.org/10.1007/s10009-019-00511-9
3. Belmonte, G., Ciancia, V., Latella, D., Massink, M.: VoxLogicA: a Spatial Model Checker for Declarative Image Analysis (Extended Version). ArXiv e-prints (Nov 2018), https://arxiv.org/abs/1811.05677
4. Belmonte, G., Ciancia, V., Latella, D., Massink, M., Biondi, M., De Otto, G., Nardone, V., Rubino, G., Vanzi, E., Banci Buonamici, F.: A topological method for automatic segmentation of glioblastoma in MR FLAIR for radiotherapy - ESMRMB 2017, 34th annual scientific meeting. Magnetic Resonance Materials in Physics, Biology and Medicine **30**(S1), 437 (oct 2017). https://doi.org/10.1007/s10334-017-0634-z, https://doi.org/10.1007/s10334-017-0634-z
5. Belmonte, G., Broccia, G., Ciancia, V., Latella, D., Massink, M.: Feasibility of spatial model checking for nevus segmentation. In: 9th IEEE/ACM International Conference on Formal Methods in Software Engineering, FormaliSE@ICSE 2021, Madrid, Spain, May 17-21, 2021. pp. 1–12. IEEE (2021).

https://doi.org/10.1109/FormaliSE52586.2021.00007, https://doi.org/10.1109/FormaliSE52586.2021.00007

6. Belmonte, G., Ciancia, V., Latella, D., Massink, M.: Innovating medical image analysis via spatial logics. In: ter Beek, M.H., Fantechi, A., Semini, L. (eds.) From Software Engineering to Formal Methods and Tools, and Back - Essays Dedicated to Stefania Gnesi on the Occasion of Her 65th Birthday. Lecture Notes in Computer Science, vol. 11865, pp. 85–109. Springer (2019). https://doi.org/10.1007/978-3-030-30985-5_7, https://doi.org/10.1007/978-3-030-30985-5_7
7. Belmonte, G., Ciancia, V., Latella, D., Massink, M.: Voxlogica: A spatial model checker for declarative image analysis. In: Vojnar, T., Zhang, L. (eds.) Tools and Algorithms for the Construction and Analysis of Systems - 25th International Conference, TACAS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings, Part I. Lecture Notes in Computer Science, vol. 11427, pp. 281–298. Springer (2019). https://doi.org/10.1007/978-3-030-17462-0_16, https://doi.org/10.1007/978-3-030-17462-0_16
8. Bezhanishvili, N., Ciancia, V., Gabelaia, D., Grilletti, G., Latella, D., Massink, M.: Geometric Model Checking of Continuous Space. Log. Methods Comput. Sci. **18**(4), 7:1–7:38 (2022), https://lmcs.episciences.org/10348, DOI 10.46298/LMCS-18(4:7)2022. Published on line: Nov 22, 2022. ISSN: 1860-5974
9. Bray, T.: The javascript object notation (json) data interchange format. https://tools.ietf.org/html/rfc8259 (2017), rFC 8259
10. Broccia, G., Ciancia, V., Latella, D., Massink, M.: Towards a GUI for declarative medical image analysis: Cognitive and memory load issues. In: Stephanidis, C., Antona, M., Ntoa, S. (eds.) HCI International 2022 Posters - 24th International Conference on Human-Computer Interaction, HCII 2022, Virtual Event, June 26 - July 1, 2022, Proceedings, Part II. Communications in Computer and Information Science, vol. 1581, pp. 103–111. Springer (2022). https://doi.org/10.1007/978-3-031-06388-6_14, https://doi.org/10.1007/978-3-031-06388-6_14
11. Bussi, L., Ciancia, V., Gadducci, F.: Towards a spatial model checker on GPU. In: Peters, K., Willemse, T.A.C. (eds.) Formal Techniques for Distributed Objects, Components, and Systems - 41st IFIP WG 6.1 International Conference, FORTE 2021, Held as Part of the 16th International Federated Conference on Distributed Computing Techniques, DisCoTec 2021, Valletta, Malta, June 14-18, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12719, pp. 188–196. Springer (2021). https://doi.org/10.1007/978-3-030-78089-0_12, https://doi.org/10.1007/978-3-030-78089-0_12
12. Bussi, L., Ciancia, V., Gadducci, F., Latella, D., Massink, M.: Towards model checking video streams using voxlogica on GPUs. In: Bowles, J., Broccia, G., Pellungrini, R. (eds.) From Data to Models and Back - 10th International Symposium, DataMod 2021, Virtual Event, December 6-7, 2021, Revised Selected Papers. Lecture Notes in Computer Science, vol. 13268, pp. 78–90. Springer (2021). https://doi.org/10.1007/978-3-031-16011-0_6, https://doi.org/10.1007/978-3-031-16011-0_6
13. Cavoukian, A., Harbour, P., Information, Commissioner/Ontario, P.: Privacy by Design in Law, Policy and Practice: A White Paper for Regulators, Decision-makers and Policy-makers. Information and Privacy Commissioner of Ontario, Canada (2011), https://books.google.it/books?id=8K9XAQAACAAJ
14. Choplin, R.H., Boehme, J.M., Maynard, C.D.: Picture archiving and communication systems: an overview. RadioGraphics **12**(1), 127–129 (1992). https:

//doi.org/10.1148/radiographics.12.1.1734458, https://doi.org/10.1148/radiographics.12.1.1734458, pMID: 1734458
15. Ciancia, V., Gilmore, S., Latella, D., Loreti, M., Massink, M.: Data verification for collective adaptive systems: Spatial model-checking of vehicle location data. In: Eighth IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops, SASOW. pp. 32–37. IEEE Computer Society (2014). https://doi.org/10.1109/SASOW.2014.16
16. Ciancia, V., Grilletti, G., Latella, D., Loreti, M., Massink, M.: An experimental spatio-temporal model checker. In: Software Engineering and Formal Methods - SEFM 2015 Collocated Workshops. Lecture Notes in Computer Science, vol. 9509, pp. 297–311. Springer (2015). https://doi.org/10.1007/978-3-662-49224-6_24
17. Ciancia, V., Latella, D., Loreti, M., Massink, M.: Specifying and verifying properties of space. In: Theoretical Computer Science - 8th IFIP TC 1/WG 2.2 International Conference, TCS 2014, Rome, Italy, September 1-3, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8705, pp. 222–235. Springer (2014). https://doi.org/10.1007/978-3-662-44602-7_18
18. Ciancia, V., Latella, D., Loreti, M., Massink, M.: Model Checking Spatial Logics for Closure Spaces. Logical Methods in Computer Science **Volume 12, Issue 4** (Oct 2016). https://doi.org/10.2168/LMCS-12(4:2)2016, http://lmcs.episciences.org/2067
19. Ciancia, V., Belmonte, G., Latella, D., Massink, M.: A hands-on introduction to spatial model checking using voxlogica - - invited contribution. In: Laarman, A., Sokolova, A. (eds.) Model Checking Software - 27th International Symposium, SPIN 2021, Virtual Event, July 12, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12864, pp. 22–41. Springer (2021). https://doi.org/10.1007/978-3-030-84629-9_2, https://doi.org/10.1007/978-3-030-84629-9_2
20. Ciancia, V., Latella, D., Massink, M., Paškauskas, R.: Exploring spatio-temporal properties of bike-sharing systems. In: 2015 IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops, SASO Workshops 2015, Cambridge, MA, USA, September 21-25, 2015. pp. 74–79. IEEE Computer Society (2015), https://doi.org/10.1109/SASOW.2015.17
21. Ciancia, V., Latella, D., Massink, M., Paškauskas, R., Vandin, A.: A tool-chain for statistical spatio-temporal model checking of bike sharing systems. In: Margaria, T., Steffen, B. (eds.) Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques - 7th International Symposium, ISoLA 2016, Imperial, Corfu, Greece, October 10-14, 2016, Proceedings, Part I. Lecture Notes in Computer Science, vol. 9952, pp. 657–673 (2016), https://doi.org/10.1007/978-3-319-47166-2_46
22. Cohn, A., Hazarika, S.: Qualitative spatial representation and reasoning: An overview. Fundamenta Informaticae **46**, 1–29 (04 2001)
23. Dragoni, N., Giallorenzo, S., Lafuente, A.L., Mazzara, M., Montesi, F., Mustafin, R., Safina, L.: Microservices: Yesterday, Today, and Tomorrow, pp. 195–216. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-67425-4_12, https://doi.org/10.1007/978-3-319-67425-4_12
24. Galton, A.: The mereotopology of discrete space. In: Spatial Information Theory. Cognitive and Computational Foundations of Geographic Information Science, Lecture Notes in Computer Science, vol. 1661, pp. 251–266. Springer (1999), http://dx.doi.org/10.1007/3-540-48384-5_17
25. Galton, A.: A generalized topological view of motion in discrete space. Theor. Comput. Sci. **305**((1-3)), 111–134 (2003), https://doi.org/10.1016/S0304-3975(02)00701-6

26. Galton, A.: Discrete mereotopology. In: Mereology and the Sciences: Parts and Wholes in the Contemporary Scientific Context, pp. 293–321. Springer International Publishing (2014), https://doi.org/10.1007/978-3-319-05356-1_11
27. Gordillo, N., Montseny, E., Sobrevilla, E.: State of the art survey on MRI brain tumor segmentation. Magn. Reson. Imaging. **31**(8), 1426–1438 (2013). https://doi.org/10.1016/j.mri.2013.05.002
28. Hrishikesh Jaware, T., Ramesh Patil, V., Nayak, C., Elmasri, A., Ali, N., Mishra, P.: A novel approach for brain tissue segmentation and classification in infants' MRI images based on seeded region growing, foster corner detection theory, and sparse autoencoder. Alexandria Engineering Journal **76**, 289–305 (2023). https://doi.org/doi.org/10.1016/j.aej.2023.06.040, https://www.sciencedirect.com/science/article/pii/S1110016823005082
29. Isensee, F., Jaeger, P.F., Kohl, S.A.A., Petersen, J., Maier-Hein, K.H.: nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. Nature Methods **18**(2), 203–211 (Feb 2021). https://doi.org/10.1038/s41592-020-01008-z, https://doi.org/10.1038/s41592-020-01008-z
30. Isensee, F., Wald, T., Ulrich, C., Baumgartner, M., Roy, S., Maier-Hein, K., Jaeger, P.F.: nnU-Net revisited: A call for rigorous validation in 3D medical image segmentation (2024)
31. Kleppmann, M., Wiggins, A., van Hardenberg, P., McGranaghan, M.: Local-first software: you own your data, in spite of the cloud. In: Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software. p. 154–178. Onward! 2019, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3359591.3359737, https://doi.org/10.1145/3359591.3359737
32. Nenzi, L., Bortolussi, L., Ciancia, V., Loreti, M., Massink, M.: Qualitative and quantitative monitoring of spatio-temporal properties with SSTL. Log. Methods Comput. Sci. **14**(4) (2018), https://doi.org/10.23638/LMCS-14(4:2)2018
33. Platzer, A.: Interlinking symbolic AI and subsymbolic AI. In: International Symposium on Leveraging Applications of Formal Methods, Verification and Validation, ISoLA 2024. Lecture Notes in Computer Science, vol. In this volume (2024)
34. Randell, D.A., Landini, G., Galton, A.: Discrete mereotopology for spatial reasoning in automated histological image analysis. IEEE Trans. Pattern Anal. Mach. Intell. **35**(3), 568–581 (2013). https://doi.org/10.1109/TPAMI.2012.128, https://doi.org/10.1109/TPAMI.2012.128
35. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10674–10685 (2022). https://doi.org/10.1109/CVPR52688.2022.01042
36. Spyridon (Spyros) Bakas et al. (Ed.): 2017 international MICCAI BraTS Challenge: pre-conference proceedings (Sept 2017), https://www.cbica.upenn.edu/sbia/Spyridon.Bakas/MICCAI_BraTS/\MICCAI_BraTS_2017_proceedings_shortPapers.pdf
37. Tsigkanos, C., Nenzi, L., Loreti, M., Garriga, M., Dustdar, S., Ghezzi, C.: Inferring analyzable models from trajectories of spatially-distributed internet of things. In: 2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS). pp. 100–106 (2019). https://doi.org/10.1109/SEAMS.2019.00021
38. Tsigkanos, C., Kehrer, T., Ghezzi, C.: Modeling and verification of evolving cyber-physical spaces. In: Bodden, E., Schäfer, W., van Deursen, A., Zisman,

A. (eds.) Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2017, Paderborn, Germany, September 4-8, 2017. pp. 38–48. ACM (2017). https://doi.org/10.1145/3106237.3106299, https://doi.org/10.1145/3106237.3106299

39. Yaniv, Z., Lowekamp, B.C., Johnson, H.J., Beare, R.: SimpleITK image-analysis notebooks: a collaborative environment for education and reproducible research. Journal of Digital Imaging **31**(3), 290–303 (Jun 2018). https://doi.org/10.1007/s10278-017-0037-8, https://doi.org/10.1007/s10278-017-0037-8