TPLP **25** (4): 558–578, 2025. © The Author(s), 2025. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.

doi:10.1017/S1471068425100136 First published online 22 August 2025

Improving ASP-Based ORS Schedules through Machine Learning Predictions

PIERANGELA BRUNO and CARMINE DODARO
DeMaCS, University of Calabria, Rende, Italy
(*e-mails:* pierangela.bruno@unical.it, carmine.dodaro@unical.it)

GIUSEPPE GALATÀ
SurgiQ srl, Genova, Italy
(*e-mail:* giuseppe.galata@surgiq.com)

MARCO MARATEA
DeMaCS, University of Calabria, Rende, Italy
(*e-mail:* marco.maratea@unical.it)

MARCO MOCHI
SurgiQ srl, Genova, Italy
(*e-mail:* marco.mochi@edu.unige.it)

submitted 22 July 2025; revised 22 July 2025; accepted 27 July 2025

Abstract

The operating room scheduling (ORS) problem deals with the optimization of daily operating room surgery schedules. It is a challenging problem subject to many constraints, like to determine the starting time of different surgeries and allocating the required resources, including the availability of beds in different department units. Recently, solutions to this problem based on answer set programming (ASP) have been delivered. Such solutions are overall satisfying but, when applied to real data, they can currently only verify whether the encoding aligns with the actual data and, at most, suggest alternative schedules that could have been computed. As a consequence, it is not currently possible to generate provisional schedules. Furthermore, the resulting schedules are not always robust. In this paper, we integrate inductive and deductive techniques for solving these issues. We first employ machine learning algorithms to predict the surgery duration, from historical data, to compute provisional schedules. Then, we consider the confidence of such predictions as an additional input to our problem and update the encoding correspondingly in order to compute more robust schedules. Results on historical data from the ASL1 Liguria in Italy confirm the viability of our integration.

KEYWORDS: healthcare application, answer set programming, neuro-symbolic approach, machine learning, operating room scheduling

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

1 Introduction

The operating room scheduling (ORS) problem consists of optimizing daily surgical schedules in operating rooms (ORs). It is a complex and highly constrained problem that requires, among other tasks, determining the starting times of surgeries and allocating the necessary resources (Meskens *et al*. 2013; Aringhieri *et al*. 2015; Abedini *et al*. 2016; Hamid *et al*. 2019). Recently, several solutions based on answer set programming (ASP) for the ORS problem have been proposed (Dodaro *et al*. 2022a, 2024), showing promising results in finding feasible and efficient schedules under realistic constraints. However, when applied to real-world data, such as those provided by ASL1 Liguria in Italy (Dodaro *et al*. 2024), these solutions rely on the assumption that surgery durations are known in advance. Specifically, since the scheduling was performed on past surgeries, the actual durations were already available. This allowed researchers to compare the ASP-generated schedules with those historically adopted by the hospital and evaluate potential improvements retrospectively. Nevertheless, in a practical setting, surgery durations are not known beforehand, and this uncertainty poses a critical challenge: ASP systems heavily depend on accurate input values, and imprecise duration estimations can lead to suboptimal scheduling solutions. As a result, the ability to generate provisional schedules under uncertainty remains largely unaddressed.

To overcome this limitation, it is necessary to integrate predictive models capable of estimating surgery durations before the actual scheduling process takes place. Machine learning (ML) techniques offer a promising solution for this purpose, enabling the estimation of surgery durations based on historical patient and surgery data.

The integration of deductive (logic-based) and inductive (ML-based) approaches has emerged as one of the most active areas of research in the AI community in recent years, and ASP is no exception. Indeed, several efforts have been made in this direction, such as using ML to guide the heuristics of ASP solvers to improve performance (Balduccini 2011; Dodaro *et al*. 2022b; Liu *et al*. 2022), applying algorithm selection techniques (Maratea *et al*. 2014; Hoos *et al*. 2014), representing and explaining ML models via ASP (Giordano and Dupré 2022; Eiter *et al*. 2023), and developing languages and tools for learning ASP programs (Yang *et al*. 2020; Law *et al*. 2020; Tarzariol *et al*. 2023; Cunnington *et al*. 2023). In addition, there has been growing interest in neuro-symbolic approaches in real-world applications where ASP is applied in conjunction with ML techniques (Eiter *et al*. 2022; Bruno *et al*. 2022; Barbara *et al*. 2023).

In this paper, we contribute to this line of research by proposing a hybrid approach that integrates ML predictions into an ASP-based solution for the ORS problem. Specifically, our contributions are as follows. First, we perform an analysis of the available real-world dataset, identifying significant distribution skewness that could negatively affect predictive accuracy. To mitigate this, we apply a dedicated preprocessing phase to improve data quality and reliability. Then, we systematically evaluate several state-of-the-art ML algorithms for predicting surgery durations, using standard performance metrics such as mean absolute error, root mean squared error, and coefficient of determination. Among the tested models, XGBoost (Chen and Guestrin 2016) achieves the best performance and is selected for further integration. Subsequently, we introduce the notion of prediction confidence by clustering the predicted durations into four discrete levels, ranging from high confidence to very low confidence, providing an additional layer of information to

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

assess prediction reliability. Then, we extend the original ASP encoding to incorporate confidence information into the scheduling process, enabling the computation of more robust and reliable surgical schedules. Finally, we conduct an extensive experimental evaluation. Despite the challenges posed by the inherent distribution skewness in the dataset, and the limited predictive accuracy of the models, the results show that incorporating ML predictions, especially when combined with confidence information, leads to a good improvement in scheduling quality. In fact, our approach obtains a better OR usage and reduces the incidence of overbooking compared to the baseline ASP encoding that relies only on statistical averages.

2 Problem and data description

In this section, we present a high-level description of the ORS problem considered in this paper and the available dataset.

The ORS problem specifications described in the following were defined by ASL1 Liguria, a local health authority in Italy that includes three hospitals: Bordighera, Sanremo, and Imperia. Such hospitals serve a population of around 213, 000 people. The central element of the ORS problem is the concept of a *registration*. Each registration represents a surgical procedure requested by a patient and is associated with a specific duration, a reservation number, a medical specialty, and a type of hospitalization. The set of registrations that have not yet been performed constitutes the surgical waiting list. The overall goal of the ORS problem is to assign as many registrations as possible from a large waiting list to appropriate ORs. Due to resource or specialty constraints, it may not be possible to assign certain surgical specialties to specific ORs. Therefore, the objective is to maximize the usage of OR time, which is an extremely valuable resource. Indeed, OR costs are estimated to be in the range of tens of dollars per minute (Smith *et al*. 2022), with approximately half representing fixed costs incurred even when the OR is not used (Macario 2010). Since patients cannot overlap within the same OR and OR overload must be avoided, the first requirement is to ensure that the total duration of surgeries assigned to any given OR does not exceed its available operating time. In the context of the three hospitals managed by ASL1 Liguria, Bordighera has two ORs available from 07:30 A.M. to 01:30 P.M., while Imperia and Sanremo each have five ORs available from 07:30 A.M. to 08:00 P.M. The ORS problem also includes aspects related to patient prioritization and OR usage. In particular, registrations may correspond to different clinical conditions and, in general, have been placed on the waiting list for varying lengths of time. These factors are jointly abstracted into a unified *priority* metric that guides the scheduling process, where registrations with the highest priority refer to patients already preplanned by the hospital and are therefore subject to a hard constraint, that is they must be scheduled, while other registrations are scheduled according to OR capacity and subject to a hierarchical preference. Furthermore, some ORs are designated for limited elective use due to their partial reservation for emergency procedures or other institutional needs.

More precisely, given a set of surgery registrations (each consisting of a patient ID, a priority level from p_1 to p_4, the required specialty, and the expected surgery duration) and a Master Surgical Schedule (MSS), which defines the specialty assigned to each OR

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

during each shift of the week, the goal is to assign each registration (i.e., each patient) to an OR, within a specific shift on a given day of the scheduling period, subject to the following constraints:

- Each registration is assigned at most once;
- The total length of the surgeries assigned to a given OR and shift must not exceed the length of the shift;
- Registrations with priority level p_1 must be assigned to an OR within the scheduling period;
- Unassigned registrations with priority levels p_2, p_3, and p_4 should be minimized, giving precedence to higher-priority cases;
- A specific OR, referred to as *OR A*, can be assigned to at most one patient, as it is reserved for emergencies.

As for the datasets, we used data taken from a weekly schedule of surgeries across the three hospitals of ASL1, as well as data from historical other weeks, including a list of available ORs for all hospitals. We collected and prepared the data for testing by working with four different files, where each file represents a different type of data. In particular, the first file contains the operating list of the considered week of surgeries, from 04/03/2019 to 10/03/2019, providing information on the required surgery, the OR, and the specialty originally scheduled. The second file includes the historical list of surgeries scheduled in 2019, which includes information on the required surgery, the starting and ending time of the surgery, and the date of the surgery. The third file includes the list of ORs in each hospital and their opening hours. The fourth file contains the list of patients hospitalized the week before the considered week of the scheduling, along with their admission and discharge times. Overall, the dataset contains 32 features. Each feature represents a different attribute related to the surgical procedure, patient, diagnosis, timing, or logistics. Each row corresponds to a single surgical intervention. Specifically, Table 1 shows all the features included in the dataset along with the corresponding description.

3 Prediction methodology and experimental evaluation

This section presents the preprocessing applied to the dataset (Section 3.1), the ML algorithms employed to analyze the data (Section 3.2), quantitative model performances (Section 3.3), and the results of the experiments in terms of confidence (Section 3.4).

3.1 Data preprocessing and distribution analysis

Data preprocessing is a crucial step in ML, particularly when working with real-world medical datasets that often suffer from skewed distributions, noise, and sparsity. In our study, the target variable, `DURATA`, representing the total operative time in minutes, was derived from the difference between `USCITASALA` (exit from OR) and `INGRESSOSALA` (entry). Both fields were parsed as `datetime` objects, and duration was computed as the difference in minutes. Entries with negative or zero durations, likely due to data entry errors, were removed from the dataset to ensure data integrity and modeling accuracy. As the goal of our study is regression, addressing target distribution characteristics is

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

Table 1. *Description of the features in the surgical procedures dataset*

Feature	Description
PROGRESSIVO	Unique identifier for each surgical record.
TIPORICOVERO	Type of hospitalization (e.g., emergency, urgent post-operative).
SESSO	Biological sex of the patient (F = Female, M = Male).
ETA	Patient age at the time of procedure, expressed in years.
REPARTO	Hospital department or unit where the procedure was performed.
PRES ANESTES	Presence of anaesthesiologist during the procedure (Yes/No).
STAMP	Internal administrative flag.
CC	Internal compliance tracking flag.
CA	Additional clinical flag.
ANESTLOC	Whether localized anesthesia was used.
DIAGNOSI1	Primary diagnostic code (ICD format).
DESCDIAGNOSI1	Description of the primary diagnosis.
INGRESSOSALA	Timestamp of patient entry into the operating room.
USCITASALA	Timestamp of patient exit from the operating room.
REGRICOVERO	Type of hospital admission (e.g., ordinary, day surgery).
CHIRURGHI_1	Identifier for the surgeon or surgical team.
ICD1	Code for the surgical procedure (ICD format).
DESCICD1	Description of the surgical procedure.
BLOCCO	Surgical block location in the hospital.
DATANASCITA	Patient date of birth.
NOSOLOGICO	Internal hospital case reference number.
DATAINTERVENTO	Date of surgical intervention.
SALA	Operating room designation.
TIPOANESTESIA	Type of anaesthesia administered.
INGRESSOBLOCCOOP	Timestamp of entry into the operative block.
PREPARAZIONEPAZIENTE	Start time of preoperative patient preparation.
INIZIOANESTESIA	Start time of anesthesia administration.
INIZIOINTERVENTO	Start time of the surgical procedure.
FINEINTERVENTO	End time of the surgical procedure.
FINEASSANESTINSALA	End of anesthesiology assistance in the operating room.
USCITABLOCCOOP	Timestamp of patient exit from operative block.
DURATA	Total procedure duration in minutes.

particularly important. As shown in Figure 1, the original distribution of the intervention durations was highly skewed, with a large concentration of cases around 15 minutes and a long tail of less frequent, prolonged procedures. This distribution skewness can significantly affect model performance, leading to biased predictions and poor generalization, particularly for minority classes or rare cases. To address these issues, we employed the following preprocessing steps:

1. Diagnoses that appeared only once within each department (REPARTO) were identified and grouped by K-Means clustering (with up to 3 clusters). This step reduced data sparsity and helped generalize rare cases by associating them with similar groups according to the department.

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

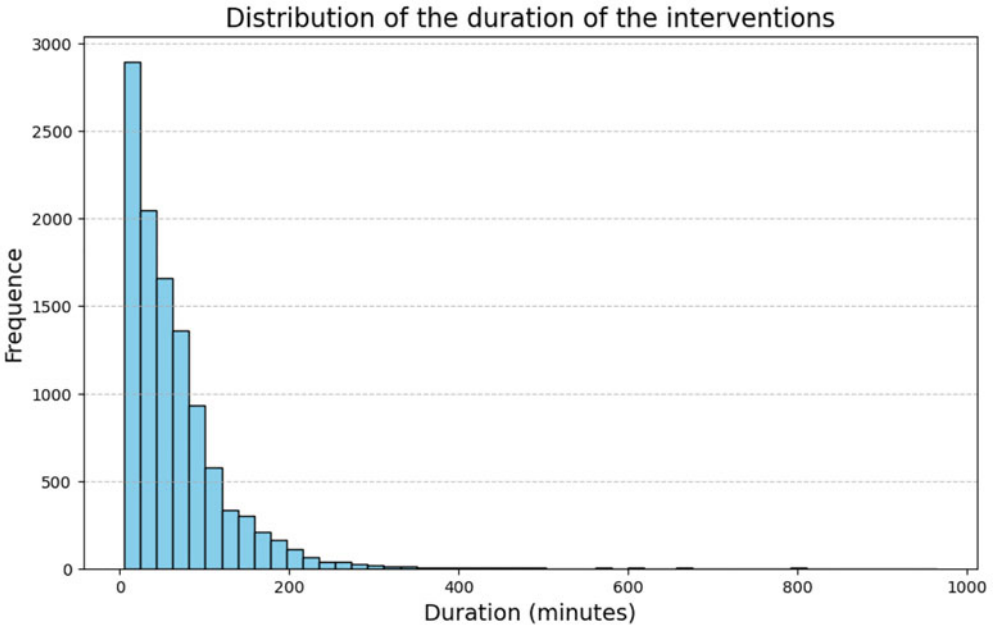


Fig. 1. Histogram of intervention durations (before preprocessing).

2. We filtered out extreme values that lay outside the typical range of durations using the Interquartile Range (IQR) method, that is a statistical measure of dispersion, calculated as the difference between the third quartile (Q3) and the first quartile (Q1) of the data (i.e., IQR = Q3 – Q1) (Dekking *et al.* 2005). This improved the central tendency and distribution spread, enhancing the learning stability of regression models.
3. To mitigate issues related to multicollinearity, we removed features that exhibited high pairwise correlation (above a 0.95 threshold). Redundant features can introduce noise, inflate model complexity, and impair generalization. By identifying and eliminating highly correlated variables, we reduced dimensionality and improved model robustness and interpretability. Indeed, by removing highly correlated features, we aimed to reduce redundancy in the dataset. When two or more features are strongly correlated, they tend to capture the same underlying information, making it difficult to isolate their individual contributions to the model. This redundancy complicates interpretation especially when assessing which variables are truly driving predictions, and reduces model stability. As discussed in Kuhn *et al.* (2013), eliminating one of several highly correlated features typically does not impair predictive performance and can lead to a simpler, more interpretable model.

In particular, Figure 2 illustrates the duration distribution after Step 2 of preprocessing. Compared to the raw dataset, the cleaned data shows a more compact distribution with fewer extreme outliers and a less pronounced right skew. Furthermore, the preprocessing performed in Step 3 resulted in a reduction of feature dimensionality from 32 to 23, reflecting the elimination of less informative features.

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

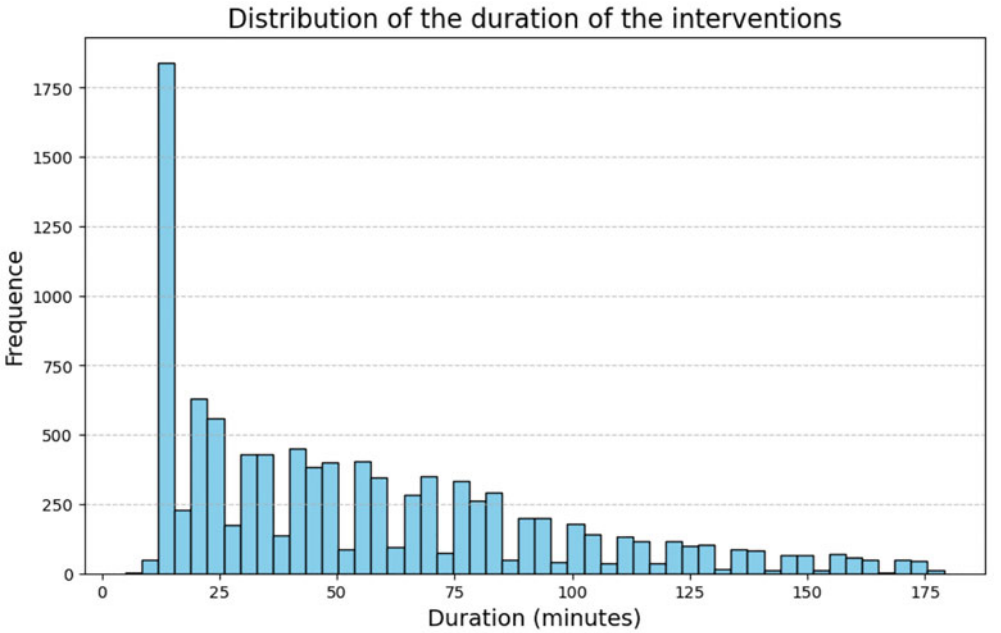


Fig. 2. Histogram of intervention durations (after preprocessing).

3.2 Predictive modeling

To predict surgical procedure durations, several ML regressors were implemented and compared. All models were trained and evaluated under identical preprocessing and evaluation pipelines to ensure fairness. The dataset was split into training and testing subsets (80 % and 20 %, respectively) using stratified sampling based on procedure duration. This strategy ensured that the distribution of durations, particularly for rare or long procedures, was preserved across both sets. In this way, we avoided scenarios where specific duration ranges (e.g., particularly long procedures) appeared only in the test set, which could otherwise result in biased or unreliable performance estimations. The following algorithms were explored:

- **Decision Tree Regressor (DT):** a simple, interpretable model that recursively splits the data based on feature thresholds to minimize prediction error (Breiman *et al.* 1984).
- **Random Forest Regressor (RF):** an ensemble method that constructs multiple decision trees using bootstrap sampling and random feature selection at each split (Breiman 2001). Final predictions are computed as the average of individual tree predictions.
- **Gradient Boosting Regressor (GB):** a sequential ensemble technique where each new tree is trained to correct the residuals of the previous ensemble (Friedman 2001). It combines weak learners into a strong predictor and includes hyperparameters for controlling learning rate, tree depth, and regularization.

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

- **Extreme Gradient Boosting (XGBoost):** a highly optimized and regularized gradient boosting framework (Chen and Guestrin 2016), known for its scalability and efficiency. XGBoost introduces system-level optimizations (e.g., parallelization) and algorithmic enhancements like shrinkage and sparsity-aware split finding.
- **K-Nearest Neighbors Regressor (KNN):** a non-parametric method that predicts the target as the average of the k nearest neighbors in the feature space (Altman 1992).
- **Support Vector Regressor (SVR):** a margin-based regression technique that aims to find a function within an ϵ-tube from the true outputs, penalizing predictions only when errors exceed ϵ (Drucker *et al*. 1996).

3.2.1 Deep learning (DL) models

DL models are generally not well-suited for tabular data, as highlighted by Shwartz-Ziv and Armon (2022) and by Borisov *et al*. (2024). Nevertheless, some of the recent DL approaches obtained good performance on tabular data, as the one proposed by Arik and Pfister (2021), called TabNet. Indeed, TabNet is a deep neural architecture tailored for tabular data that uses sequential attention to select features at each decision step, enabling both high performance and interpretability. Motivated by this promising finding, we tested the performance of TabNet on our dataset and we adapted and evaluated the classical DL models **Multi-Layer Perceptron (MLP) (Almeida 2020) and 1D Convolutional Neural Network (1D-CNN)** (Ige and Sibiya 2024). Each model was optimized via grid search and the best configuration is reported in Table 2. In more detail, MLP was implemented as a feedforward neural network with the Rectified Linear Unit (ReLU), a nonlinear activation function that introduces nonlinearity into the model and helps mitigate the vanishing gradient problem. It outputs the input directly if it is positive and zero otherwise, effectively retaining only the positive part of its argument. We also used mean squared error loss and adaptive moment estimation (Adam) optimizer to make training faster and more stable, especially on noisy or sparse data (Kingma and Ba 2015). Our implementation followed a standard configuration with two hidden layers and 64 units per layer. 1D-CNN was adapted to tabular input by reshaping the feature vectors as sequences, applying a single convolutional layer with 32 filters and kernel size 2, followed by a fully connected hidden layer and regression output. TabNet was tested using a flexible wrapper that allowed us to tune key architectural parameters such as the size of internal layers, the number of steps in the attention process, and the strength of feature sparsity. Unlike standard neural networks, TabNet learns which features to use at each step, making it more efficient for tabular tasks.

3.3 Evaluation metrics and experiments

To evaluate the performance of the ML and DL models in predicting surgical procedure durations, we employed three standard regression metrics. The first one is the **mean absolute error (MAE)** that reflects the average magnitude of prediction errors, and is computed as the average of the absolute differences between the predicted values $\hat{y}_i$ and

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

Table 2. *Best hyperparameter configuration for each algorithm*

Algorithm	Hyperparameter
DT	`max_depth` = 50, `min_samples_split` = 2, `criterion`=`friedman_mse`
RF	`n_estimators` = 10, `max_depth`=`None`, `min_samples_split` = 5
GB	`n_estimators` = 400, `learning_rate` = 0.01, `max_depth` = 15
XGBoost	`n_estimators` = 400, `learning_rate` = 0.1, `max_depth` = 5
KNN	`n_neighbors` = 5, `weights`=`distance`
SVR	`C` = 1, `epsilon` = 0.01, `kernel`=`rbf`
MLP	`epochs` = 100, `batch_size` = 64
1D-CNN	`epochs` = 50, `batch_size` = 8
TabNet	`n_d` = 8, `n_a` = 16, `n_steps` = 3, `gamma` = 2.0, `lambda_sparse` = 0.0001, `lr` = 0.01

Table 3. *Model performance using best parameters. Best results are in bold*

Model	MAE (min)	RMSE (min)	R^2
DT	16.99	26.74	0.55
RF	12.93	19.45	0.76
GB	13.32	20.42	0.74
XGBoost	**12.36**	**18.20**	**0.79**
KNN	15.69	24.03	0.64
SVR	17.22	26.87	0.55
MLP	15.73	23.28	0.66
1D-CNN	15.24	22.57	0.68
TabNet	14.45	22.38	0.69

the true values y_i, that is $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$. The second metric is the **root mean squared Error (RMSE)**, which penalizes larger errors more than **MAE** and reflects the model's sensitivity to outliers. RMSE is calculated as the square root of the mean squared error, that is $\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$. The last one is the **coefficient of determination (R^2)**, which measures the proportion of variance in the target variable explained by the model, that is $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$, where $\bar{y}$ is the mean of the observed values. In addition to model evaluation, we conducted hyperparameter tuning using grid search with cross-validation. The final configuration for each model was selected based on the combination of hyperparameters that achieved the lowest MAE and RMSE averaged across the validation folds during cross-validation, as shown in Table 2 (for details about the hyperparameters see https://github.com/DeMaCS-UNICAL/ML4ORS). Moreover, Table 3 highlights the performance of each ML model in predicting surgical procedure durations. Among all candidates, the XGBoost Regressor achieved the best results, with the lowest MAE (12.36 minutes), lowest RMSE (18.20 minutes), and highest R^2 score (0.79). RF and GB followed closely behind, also demonstrating strong predictive performance with slightly higher error metrics. In contrast, simpler models such as DT, KNN, and SVR showed substantially lower performance. The DT and KNN regressors yielded

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

higher MAE values (16.99 and 15.69 minutes, respectively) and lower R^2 scores (0.55 and 0.64), indicating limited generalization. The SVR model performed comparably to the DT, with the highest MAE (17.22 minutes) and RMSE (26.87 minutes), and an R^2 of 0.55, suggesting both struggled to capture the underlying structure of the data.

These discrepancies in performance can be partially explained by the distributional characteristics of the target variable. As shown in Figure 2, the duration data remains skewed even after preprocessing. Most surgeries are short-clustered around 15–20 minutes, but a long tail of high-duration outliers exists. This distribution skewness makes the prediction task inherently challenging, particularly for models like SVR and KNN, which are more sensitive to variance and do not adapt well to skewed or noisy distributions. In contrast, ensemble methods like XGBoost and GB are well-suited to handle skewed data, thanks to their iterative learning approach and ability to capture complex, nonlinear relationships. It is important to observe that the overall error is still non-negligible (as it is around 12 minutes on average). Nevertheless, this level of accuracy, combined with the confidence estimation framework (where over 65 % of predictions were classified as High or Moderate confidence) makes the models suitable for potential deployment in time-sensitive hospital operations and to be employed to improve the robustness of the schedules, as we will show in Section 4.3.

Concerning DL approaches, it is possible to observe that they do not outperform XGBoost in this task. However, TabNet demonstrates improved performance compared to other DL models, with an MAE of 14.45 minutes, RMSE of 22.38 minutes, and R^2 of 0.69, which is better than both the 1D-CNN (MAE: 15.24, RMSE: 22.57, R^2: 0.68) and MLP (MAE: 15.73, RMSE: 23.28, R^2: 0.66). Nevertheless, despite its competitive performance, TabNet still remains slightly behind XGBoost (MAE: 12.36, RMSE: 18.20, R^2: 0.79).

Moreover, to interpret the contribution of each feature to the model's predictions, we employed SHAP (SHapley Additive exPlanations) that assigns each feature an importance value for a particular prediction (Lundberg and Lee 2017). Figure 3 shows the SHAP summary plot for the best-performing model (XGBoost), where features are ranked by their overall contribution to the predictions. The horizontal spread reflects the variability in feature impact across all samples, while the color gradient indicates the feature value (from low in black to high in red). As an example, `REGRICOVERO`, representing the type of hospital admission, shows a wide range of SHAP values, highlighting its strong and variable influence on predicted operative duration. Higher values of this feature (in red) tend to increase the predicted duration, whereas lower values (in blue) are associated with shorter procedures. This suggests the model has learned to associate certain admission types (e.g., complex or urgent cases) with longer surgery times.

3.4 Confidence estimation

To assess the precision of individual predictions, we computed the **absolute percentage error (APE)** for each instance. This metric quantifies the relative difference between the predicted value $\hat{y}$ and the true target value y:

$$\text{APE} = \frac{|\hat{y} - y|}{y} \cdot 100.$$

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

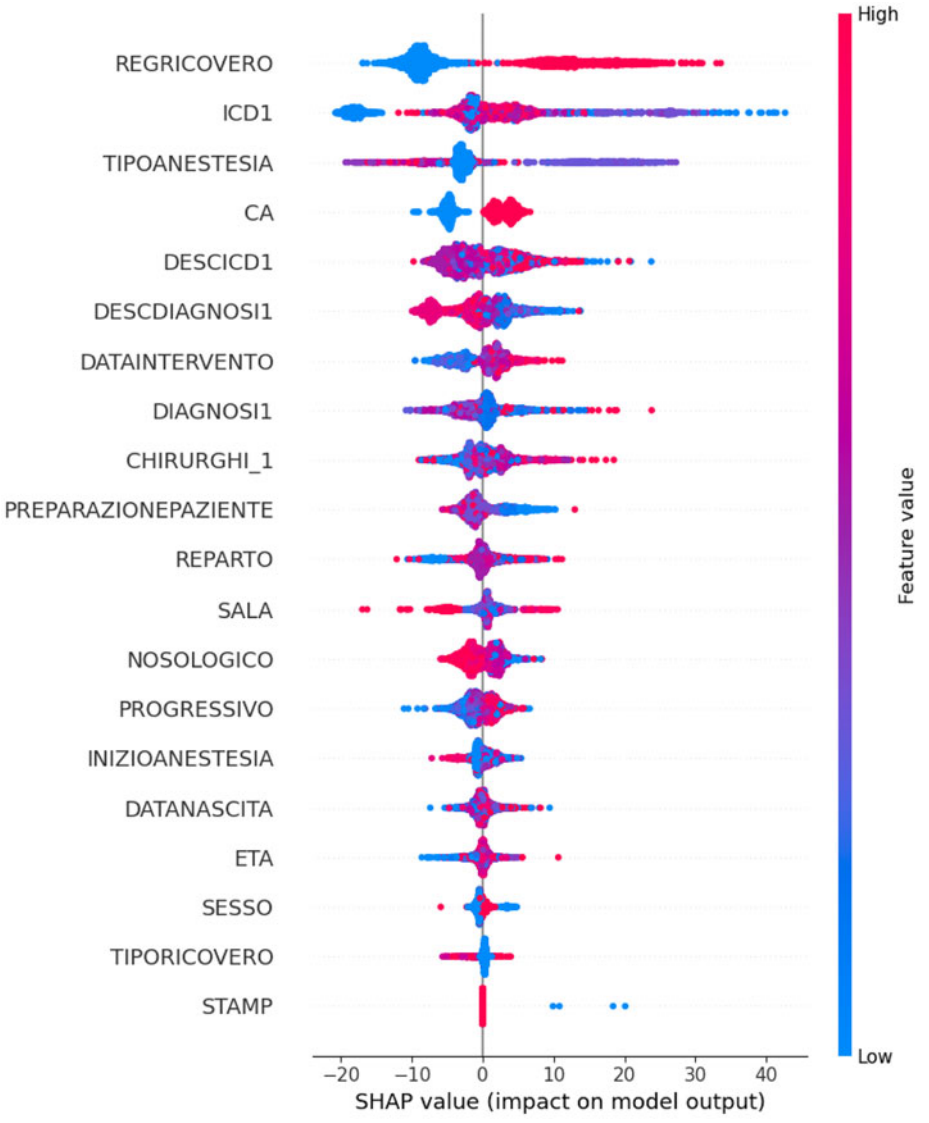


Fig. 3. SHAP summary plot for the best-performing regression model (XGBoost).

In particular, the absolute value ensures that the error is always positive, and the percentage form allows for intuitive interpretation across varying scales.

Based on the APE, we categorized each prediction into one of four confidence levels, namely **high confidence** when APE is less than 10 %, **moderate confidence** when APE is between 10 % and 25 %, **low confidence** when APE is between 25 % and 50 %, and **very low confidence** when APE is greater than or equal to 50 %.

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

```
1 {x(ID,P,OR,DAY,SHIFT)} :- registration(ID,P,SP,DUR), mss(OR,SP,SHIFT,DAY).
2 :- #count{OR,DAY,SHIFT: x(ID,P,OR,DAY,SHIFT)} > 1, registration(ID,_,_,_).
3 :- mss(OR,_,SHIFT,DAY), #sum{DUR, ID: x(ID, _, OR, DAY, SHIFT),
      registration(ID,_,_,DUR)} > DURATION, shift(SHIFT, DURATION).
4 :- not x(ID,_,_,_,_), registration(ID,1,_,_).
5 :~ not x(ID,_,_,_,_), registration(ID,P,_,_). [1@7-P, ID]
6 :- #count{ID: x(ID,_,_,"OR A",_)} > 1.
```

Listing 1. ASP encoding for the ORS problem.

4 Improving ASP-based ORS solution through ML predictions

In this section, we first review the ASP encoding of the ORS problem evaluated on real data employed by Dodaro *et al.* (2024) (Section 4.1). Then, we present the changes needed to take into account ML predictions (Section 4.2). Finally, we show the results of an experimental analysis that demonstrate the improvements of our approach (Section 4.3).

In the following sections, we assume the reader is familiar with logic programming conventions, and with ASP syntax and semantics (Brewka *et al.* 2011; Calimeri *et al.* 2020).

4.1 ASP-based ORS solution

In this section, we briefly describe an existing ASP-based encoding for the ORS problem. We begin by describing the problem's input and expected output and then show the core rules that model the scheduling constraints and objectives within the ASP framework. This encoding serves as the baseline for the extensions introduced in the following sections.

4.1.1 Data model

The input data is specified by means of the following constants and atoms. Instances of `registration(ID,P,SP,DUR)` represent the registration of the patient identified by an ID (`ID`) with priority level (`P`), the requested specialty (`SP`), and the expected duration of the surgery (`DUR`). Instances of `mss(OR,SP,SHIFT,DAY)` represent which specialty (`SP`) is assigned to an OR (`OR`) in a shift (`SHIFT`) on a day (`DAY`). Instances of `shift(SHIFT,DURATION)` indicate that the total duration of all surgeries scheduled in shift `SHIFT` must not exceed `DURATION` (expressed in time slots). The output is represented by atoms of the form `x(ID,P,OR,DAY,SHIFT)`, whose intuitive meaning is that the patient identified by an ID (`ID`) having a priority (`P`) is assigned to the OR (`OR`) in the shift (`SHIFT`) on the day (`DAY`).

4.1.2 Encoding

The related encoding is shown in Listing 1. The choice rule assigns an OR, a day, a shift to each registration. The second rule ensures that each registration is assigned at most once. The third rule ensures that the total duration of the assigned registration does not exceed the length of the shift. The fourth rule ensures that every registration with

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

```
1 sumConfidence(DAY, OR, SHIFT, N) :- mss(OR, _, SHIFT, DAY), #sum{L, ID:
      confidence(ID, L), x(ID, _, OR, DAY, SHIFT)} = N.
2 maxConfidenceDay(N) :- #max{L: sumConfidence(_, _, _, L)} = N.
3 minConfidenceDay(N) :- #min{L: sumConfidence(_, _, _, L)} = N.
4 :~ maxConfidenceDay(N). [N@2]
5 :~ maxConfidenceDay(MAX), minConfidenceDay(MIN), MAX-MIN > 0. [MAX-MIN@1]
```

Listing 2. Additional rules to take into account confidence.

priority p_1 is assigned to some OR on a day, shift. The weak constraint then minimizes the number of unassigned registrations with priority p_2, p_3, and p_4. Finally, the last constraint is added to ensure that the specific OR "`OR A`" is assigned to at most one patient, as it was reserved for emergencies and used in this limited way in the original data.

4.2 Integration of the ML predictions in ASP

As discussed before, the ORS solutions described by Dodaro *et al.* (2024) can only verify whether the encoding aligns with the actual data and, at most, suggest alternative schedules that could have been computed. Thus, it is not possible to generate provisional schedules. Furthermore, the resulting schedules are not always robust. In this section, we describe how this encoding has been extended to incorporate predictive information obtained through ML techniques. The core idea is to exploit the confidence scores produced by the predictive models (specifically, the XGBoost algorithm, which achieved the best results in our evaluation, as shown in Section 3.3) in order to guide the scheduling decisions made by the ASP program.

To this end, we first update the parameter `DUR` of the registrations. Indeed, now the duration is predicted by the ML algorithm. Then, we also introduce atoms of the form `confidence(ID,L)` that collect the confidence associated to each registration, where `L` is 1 if the confidence is High, 2 if the confidence is Moderate, 3 if the confidence is Low, and 4 if the confidence is Very Low.

Finally, we handle these two changes by adding a new set of rules, reported in Listing 2. The key idea is to prefer balanced and high-confidence scheduling decisions. Indeed, the first rule derives the sum of all the confidences assigned to a particular day. The subsequent two rules derive the values corresponding to the maximum and the minimum sum of confidences in each day, respectively. Then, the first weak constraint penalizes solutions where the maximum sum of confidence scores assigned to a day and OR is high, whereas the second one promotes balance across ORs and days by penalizing the difference between the maximum and minimum confidence sums. In this way, we aim at distributing equally among the ORs and days the patients with high confidence value.

4.3 Experiments

This section presents the experimental evaluation conducted to assess the effectiveness of the proposed neuro-symbolic approach.

First of all, it is important to observe that, in the context of ORS, it is not possible to determine the exact duration of a surgical procedure in advance. Typically, the

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

parameter `DUR` of the atoms `registration(ID,P,SP,DUR)` is assigned by hospital staff based on prior experience. In most cases, it corresponds either to the average duration of surgeries of the same type or to the average duration of surgeries performed within the same department. Therefore, for the purposes of the following experimental analysis, we compare these two traditional approaches (average by procedure type and average by department) against ML-based strategies described in the previous sections. Specifically, we replace the `DUR` value in the ASP encoding with the respective predicted duration, compute the surgical schedule using ASP, and then evaluate the quality of the schedules based on the actual durations of the procedures. The real durations, available only after the scheduling has been executed, are crucial for accurately estimating how effectively the ORs were used on each day.

We evaluate all the approaches on real-world data provided by ASL1 Liguria and already used by Dodaro *et al*. (2024). The evaluation focuses on key metrics such as the OR percentage occupancy (mean, standard deviation, minimum, and maximum), and the number of times in which an OR has been underbooked (an OR is used less than 80 % of the time) or overbooked (an OR is used for more than 100 % of the time). It is important to note that, with respect to these parameters, ASP solutions (even within the same solver) may exhibit unintuitive behavior. For instance, a solution that is preferable in terms of optimality (e.g., lower cost) may still yield worse overbooking and underbooking values, since these aspects are evaluated only in a post-processing phase. To mitigate this issue, in our encoding, the weak constraints related to confidence are assigned a lower priority level than the original ones. This ensures that performance differences can be attributed primarily to the introduction of the new constraints.

The comparison has been carried out on an Apple M1 CPU machine with 8 GB of physical RAM and a time limit of 60 s per run. As ASP system, we used clingo v. 5.6.2, configured with parameters *–restart-on-model* and *–parallel-mode = 6*. These parameters have been found to be effective in a preliminary analysis we performed with several options.

The results are presented in Table 4. The column **VBA** refers to the virtually best approach, in which the durations are set to their actual values, known only after the surgeries have been performed. This serves as a reference for the optimal performance achievable by predictive methods. The column **Conf.** refers to the method in which the parameter `DUR` is set using the durations predicted by the ML model XGBoost, with the ASP encoding also taking into account the associated confidence information. The column **Pred.** corresponds to the method where `DUR` is set using the XGBoost predictions without considering the confidence scores. The columns **Dep.** and **Surg.** represent the methods where `DUR` is assigned based on the average duration per department and per surgical procedure, respectively.

For the evaluation of the scheduling results, it is important to note that the ideal situation is to maximize the usage of ORS without exceeding their available capacity. Thus, an OR occupancy rate close to, but not exceeding, 100 % is preferred. Schedules that lead to overbooking (i.e., planned durations exceeding the available OR time) are undesirable, as they can cause operational disruptions and delays. Similarly, underbooking (i.e., leaving significant unused OR time, i.e. below 80 %) should be minimized, although it is generally less critical than overbooking. Therefore, among the generated schedules, the

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

Table 4. *Comparison of the different methods*

	Bordighera					Imperia					Sanremo				
	VBA	**Conf.**	**Pred.**	**Dep.**	**Surg.**	**VBA**	**Conf.**	**Pred.**	**Dep.**	**Surg.**	**VBA**	**Conf.**	**Pred.**	**Dep.**	**Surg.**
% OR occ. (mean)	99	96	95	101	88	99	100	100	101	102	100	101	101	103	103
OR occ. (std)	0.87	11.32	12.64	11.25	10.35	0.68	10.32	11.16	13.14	10.21	0.51	10.21	7.8	13.24	8.65
% OR occ. (max)	99	124	120	132	107	99	126	152	136	132	100	118	114	120	119
% OR occ. (min)	97	79	67	88	69	97	78	80	60	81	99	86	85	74	94
Overbooking	0	9	9	11	10	0	40	40	42	44	0	5	9	8	8
Underbooking	0	1	2	0	6	0	1	1	2	0	0	0	0	1	0

most desirable are those that maintain high, balanced occupancy rates without causing overbooking and with minimal underbooking.

By analyzing the results, we observe that the method **Conf.** (XGBoost predictions with confidence information) achieves the best overall performance, followed by **Pred.** (XGBoost predictions without confidence). In Bordighera, the mean OR occupancy with **Conf.** is 96 %, which is very close to the ideal 100 % threshold, outperforming **Dep.** (101 %) and **Surg.** (88 %). Additionally, the standard deviation is reasonably low, indicating consistent scheduling performance across different days. The number of overbooked cases is slightly lower or comparable to the other methods, and underbooking is almost negligible. Also **Pred.** obtains a good performance being quite close to **Conf.**. In Imperia, all methods reach a mean occupancy around 100 %, but **Conf.** maintains better control over the standard deviation compared to **Dep.** and **Surg.**, and slightly fewer overbooked cases are observed. Similarly, in Sanremo, the **Conf.** and **Pred.** methods result in a mean occupancy closer to 100 % (101 %), whereas methods like **Dep.**, and **Surg.** tend to overbook, with mean occupancies exceeding 103 %. Furthermore, **Conf.** achieves the lowest number of overbooking (5) compared to all the other methods, and avoids underbooking altogether.

Overall, integrating confidence information into the scheduling process improves the average OR usage and also helps limiting extreme cases of overbooking and underbooking, leading to more balanced and operationally feasible schedules.

5 Related work

The integration of symbolic reasoning and ML has garnered significant attention in recent years, leading to the emergence of neuro-symbolic AI. This paradigm aims to combine the learning capabilities of neural networks with the interpretability and formal reasoning of symbolic approaches (Sheth and Roy 2024). In the context of ASP, neuro-symbolic methods have been explored to enhance various applications, including knowledge representation and visual question answering (VQA). In particular, Barbara *et al*. (2023) present a combination of deep learning techniques with ASP, and allows for identifying possible anomalies and errors in the final product of an Italian Company operating in electrical control panel production, which provided real data. Bruno *et al*. (2022) define a framework to represent and solve explicit knowledge via ASP, taking advantage of it for driving decisions taken by neural networks and refining the output for providing explanations and interpretations. The framework has been tested on semantic segmentation tasks over two datasets of biomedical images. As for VQA, Eiter *et al*. (2022) introduce a neuro-symbolic pipeline for the analysis of CLEVR, which is a well-known dataset that consists of pictures showing scenes with objects and questions related to them. They employ confidence thresholds into the logic programs to be solved by an ASP solver, with the goal of making robust VQA systems. In the same VQA domain, Riley and Sridharan (2019) and Basu *et al*. (2020) also combine ASP and ML.

Concerning the ORS problem, Şeyda Gür and Eren (2018) provide a comprehensive overview of various approaches, discussing both different solution methods and alternative problem formulations. Among the works proposing and evaluating solutions on real-world data, Aringhieri *et al*. (2015) address the scheduling of surgical interventions

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

over a one-week planning horizon, considering multiple departments sharing a fixed number of ORs and post-operative beds. The authors propose a two-phase approach aimed at minimizing patient waiting times and maximizing the usage of hospital resources. Their method first generates a feasible assignment and then optimizes the scheduling plan. Similarly, Landa *et al*. (2016) tackle the ORS problem by decomposing it into two interconnected sub-problems: assigning patients to specific dates within a given planning horizon and determining their allocation and sequencing within the ORs. To solve this, a hybrid two-phase optimization algorithm is introduced, combining neighborhood search techniques with Monte Carlo simulation to efficiently explore the solution space. Another relevant contribution is provided by Hamid *et al*. (2019), who incorporate Decision-Making Styles (DMS) of surgical teams to better handle constraints related to material and resource availability, patient priorities, and the competencies of surgical staff. They develop a multi-objective mathematical model and design two metaheuristic algorithms to find Pareto-optimal solutions, which have been validated using data collected from a hospital in Iran. In a different direction, Zhang *et al*. (2017) address the scheduling of both elective and non-elective patients by introducing a time-dependent policy that prioritizes patients dynamically based on urgency levels and waiting times. The problem is formulated as a stochastic shortest-path Markov Decision Process (MDP) with blind alleys and is solved using an asynchronous value iteration method. Experimental results on synthetic data show that the time-dependent policy significantly reduces patient waiting times compared to classical MDP models, without leading to excessive increases in OR usage. The ORS problem has also been successfully solved by ASP solutions (Dodaro *et al*. 2019) also in the presence of additional resources, such as beds (Dodaro *et al*. 2022a) and care units (Galatà *et al*. 2021), demonstrating its flexibility in modeling complex scheduling scenarios with multiple resource constraints. In this work, however, we focus on a different aspect of the problem: we build upon the real-world data presented by (Dodaro *et al*. 2024), concentrating specifically on the prediction and integration of surgical procedure durations. Additional resources, such as post-operative beds or care units, were not explicitly considered in our model, as they do not directly impact the duration of surgical interventions, that is, the key parameter that our ML models were designed to predict and integrate into the scheduling process.

Our work is based on a neuro-symbolic approach that leverages ML predictions to enhance the robustness and efficiency of ASP-based scheduling. To the best of our knowledge, this is the first paper in this direction for the ORS problem.

6 Conclusion

In this paper, we have proposed a neuro-symbolic approach to the ORS problem, combining ML techniques with ASP. Starting from an existing ASP encoding, we extended the model to incorporate the confidence scores produced by an ML predictor, specifically, an XGBoost model trained to estimate surgical intervention durations. By integrating this predictive information into the scheduling process, we aimed to enhance both the robustness and the practical efficiency of the resulting schedules. Experimental evaluations conducted on real-world data from ASL1 Liguria demonstrated that our approach achieves better OR usage compared to traditional methods based on historical averages.

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

In particular, leveraging confidence information allowed the ASP solver to generate schedules that are closer to the ideal occupancy threshold reducing overbooking or underbooking issues w.r.t. existing approaches. These results highlight the potential of neuro-symbolic techniques in improving scheduling performance in complex, resource-constrained environments. As future work, one might take into account additional predictive elements, for example patient-specific surgical risks, or apply our approach to other problems. Moreover, we also observe that the confidence levels derived from APE thresholds cannot provide uncertainty estimations at inference time, as they are computed a posteriori and require access to the ground truth values. In this paper, our objective was not to quantify model uncertainty, but rather to demonstrate how prediction quality metrics can be integrated into the scheduling optimization. Furthermore, since APE relies on ground truth values, the absence of such data may be addressed by leveraging external estimations of expected durations based on standardized clinical averages per procedure type as reported in the medical literature. Nevertheless, exploring uncertainty quantification techniques capable of providing confidence estimations directly at inference time represents a promising direction for future work. In particular, since XGBoost emerged as the best-performing approach in our evaluation, it would be interesting to investigate whether using confidence measures specifically tailored to XGBoost could further improve performance. In our preliminary analysis, the version of XGBoost combined with our confidence slightly outperformed the variants using dedicated confidence measures. However, various parameter combinations and configurations remain to be explored. While such an approach might yield better performance, it would come at the cost of reduced generality compared to the method proposed in this paper. Nonetheless, this direction represents a promising avenue for future research. All material for reproducibility is available at https://github.com/DeMaCS-UNICAL/ML4ORS.

Acknowledgments

Carmine Dodaro and Marco Maratea were supported by the European Union - NextGenerationEU and by Italian Ministry of Research (MUR) under PNRR project FAIR "Future AI Research," CUP H23C22000860006 and by the European Union – NextGenerationEU and by the Ministry of University and Research (MUR), National Recovery and Resilience Plan (NRRP), Mission 4, Component 2, Investment 1.5, project "RAISE – Robotics and AI for Socio-economic Empowerment" (ECS00000035) under the project "Gestione e Ottimizzazione di Risorse Ospedaliere attraverso Analisi Dati, Logic Programming e Digital Twin (GOLD)," CUP H53C24000400006. Carmine Dodaro and Pierangela Bruno were supported by the European Union - NextGenerationEU and by Italian Ministry of Research (MUR) under PNRR project Tech4You "Technologies for climate change adaptation and quality of life improvement," CUP H23C22000370006. The research of Marco Mochi and Giuseppe Galatà is partially funded by the "POR FESR Liguria 2014-2020."

Competing interests

The authors declare none.

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

References

Abedini, A., Ye, H. and Li, W. 2016. Operating room planning under surgery type and priority constraints. *Procedia Manufacturing* 5, 15–25.

Almeida, L. B. 2020. Multilayer perceptrons. In *Handbook of Neural Computation*. CRC Press, C1–2.

Altman, N. 1992. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician* 46, 3, 175–185.

Arik, S.Ö. and Pfister, T. 2021. TabNet: Attentive interpretable tabular learning. In AAAI. AAAI Press, 6679–6687.

Aringhieri, R., Landa, P., Soriano, P., Tànfani, E. and Testi, A. 2015. A two level metaheuristic for the operating room scheduling and assignment problem. *Computers & Operations Research* 54, 21–34.

Balduccini, M. 2011. Learning and using domain-specific heuristics in ASP solvers. *AI Communications* 24, 2, 147–164.

Barbara, V., Guarascio, M., Leone, N., Manco, G., Quarta, A., Ricca, F. and Ritacco, E. 2023. Neuro-symbolic AI for compliance checking of electrical control panels. *Theory and Practice of Logic Programming* 23, 4, 748–764.

Basu, K., Shakerin, F. and Gupta, G. 2020. AQuA: ASP-based visual question answering. In *PADL*, Springer, Vol. 12007 of LNCS, 57–72.

Borisov, V., Leemann, T., Sessler, K., Haug, J., Pawelczyk, M. and Kasneci, G. 2024. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems* 35, 6, 7499–7519.

Breiman, L. 2001. Random forests. *Machine Learning* 45, 5–32.

Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. 1984. Classification and regression. *Trees* 40, 358.

Brewka, G., Eiter, T. and Truszczynski, M. 2011. Answer set programming at a glance. *Communications of the ACM* 54, 12, 92–103.

Bruno, P., Calimeri, F. and Marte, C. 2022. DeduDeep: An extensible framework for combining deep learning and asp-based models. In *LPNMR*, Springer, Vol.13416 of *LNCS*, 505–510.

Calimeri, F., Faber, W., Gebser, M., Ianni, G., Kaminski, R., Krennwallner, T., Leone, N., Maratea, M., Ricca, F. and Schaub, T. 2020. ASP-Core-2 input language format. *Theory and Practice of Logic Programming* 20, 2, 294–309.

Chen, T. and Guestrin, C. 2016. XGBoost: A scalable tree boosting system. In *SIGKDD*, Association for Computing Machinery, 785–794.

Cunnington, D., Law, M., Lobo, J. and Russo, A. 2023. Neuro-symbolic learning of answer set programs from raw data. In *IJCAI*, ijcai.org, 3586–3596.

Dekking, F. M., Kraaikamp, C., Lopuhaä, H. P. and Meester, L. E. 2005. *A Modern Introduction to Probability and Statistics: Understanding Why and How*. In *Springer Texts in Statistics*. Springer London.

Dodaro, C., Galatà, G., Gebser, M., Maratea, M., Marte, C., Mochi, M. and Scanu, M. 2024. Operating room scheduling via answer set programming: Improved encoding and test on real data. *Journal of Logic and Computation* 34, 8, 1556–1579.

Dodaro, C., Galatà, G., Khan, M. K., Maratea, M. and Porro, I. 2022a. Operating room (re)scheduling with bed management via ASP. *Theory and Practice of Logic Programming* 22a, 2, 229–253.

Dodaro, C., Galatà, G., Maratea, M., Porro, I., Ghidini, C., Magnini and B., Passerini, A. 2019. An ASP-based framework for operating room scheduling. *Intelligenza Artificiale* 13, 1, 63–77.

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

Dodaro, C., Ilardi, D., Oneto, L. and Ricca, F. 2022b. Deep learning for the generation of heuristics in answer set programming: A case study of graph coloring. In *LPNMR*, Springer, Vol. 13416 of LNCS, 145–158.

Drucker, H., Burges, C. J., Kaufman, L., Smola, A. and Vapnik, V. 1996. Support vector regression machines. In *Advances in Neural Information Processing Systems*, MIT Press, 9.

Eiter, T., Geibinger, T., Higuera, N. and Oetsch, J. 2023. A logic-based approach to contrastive explainability for neurosymbolic visual question answering. In *IJCAI*, ijcai.org, 3668–3676.

Eiter, T., Higuera, N., Oetsch, J. and Pritz, M. 2022. A neuro-symbolic ASP pipeline for visual question answering. *Theory and Practice of Logic Programming* 22, 5, 739–754.

Friedman, J. H. 2001. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics* 29, 5, 1189–1232.

Galatà, G., Maratea, M., Mochi, M., Morozan, V. and Porro, I. 2021. An asp-based solution to the operating room scheduling with care units. In *IPS and RCRA*, vol. 3065 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Giordano, L. and Dupré, D. T. 2022. An ASP approach for reasoning on neural networks under a finitely many-valued semantics for weighted conditional knowledge bases. *Theory and Practice of Logic Programming* 22, 4, 589–605.

Hamid, M., Nasiri, M. M., Werner, F., Sheikhahmadi, F. and Zhalechian, M. 2019. Operating room scheduling by considering the decision-making styles of surgical team members: A comprehensive approach. *Computers & Operation Research* 108, 166–181.

Hoos, H. H., Lindauer, M. and Schaub, T. 2014. Claspfolio 2: Advances in algorithm selection for answer set programming. *Theory and Practice of Logic Programming* 14, 4-5, 569–585.

Ige, A. O. and Sibiya, M. 2024. State-of-the-art in 1D convolutional neural networks: A survey. *IEEE Access* 12, 144082–144105.

Kingma, D. P. and Ba, J. 2015. Adam: A method for stochastic optimization. In *ICLR*. http://arxiv.org/abs/1412.6980.

Kuhn, M., Johnson, K., Kuhn, M. and Johnson, K. 2013. *Data Pre-Processing*. Springer.

Landa, P., Aringhieri, R., Soriano, P., Tànfani, E. and Testi, A. 2016. A hybrid optimization algorithm for surgeries scheduling. *Operations Research for Health Care* 8, 103–114.

Law, M., Russo, A. and Broda, K. 2020. The ILASP system for inductive learning of answer set programs. *CoRR*. https://arxiv.org/abs/2005.00904.

Liu, L., Truszczynski, M. and Lierler, Y. 2022. A machine learning system to improve the performance of ASP solving based on encoding selection. In *LPNMR*, Springer, Vol. 13416 of *LNCS*, 415–428.

Lundberg, S. M. and Lee, S.-I. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, Vol. 30, Curran Associates Inc., 4768–4777.

Macario, A. 2010. What does one minute of operating room time cost? *Journal of Clinical Anesthesia* 22, 4, 233–236.

Maratea, M., Pulina, L. and Ricca, F. 2014. A multi-engine approach to answer-set programming. *Theory and Practice of Logic Programming* 14, 6, 841–868.

Meskens, N., Duvivier, D. and Hanset, A. 2013. Multi-objective operating room scheduling considering desiderata of the surgical team. *Decision Support Systems* 55, 2, 650–659.

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press

Riley, H. and Sridharan, M. 2019. Integrating non-monotonic logical reasoning and inductive learning with deep learning for explainable visual question answering. *Frontiers Robotics AI* 6, 125.

Şeyda, G and Eren, T. 2018. Application of operational research techniques in operating room scheduling problems: literature overview. *Journal of Healthcare Engineering* 2018, 5341394. doi: 10.1155/2018/5341394.

Sheth, A. P. and Roy, K. 2024. Neurosymbolic value-inspired artificial intelligence (why, what, and how). *IEEE Intelligent Systems* 39, 1, 5–11.

Shwartz-Ziv, R. and Armon, A. 2022. Tabular data: Deep learning is not all you need. *Information Fusion* 81, 84–90.

Smith, T., Evans, J., Moriel, K., Tihista, M., Bacak, C., Dunn, J., Rajani, R. and Childs, B. 2022. Cost of OR Time is $46.04 per Minute. *Journal of Orthopaedic Business* 2, 10–13.

Tarzariol, A., Gebser, M., Schekotihin, K. and Law, M. 2023. Learning to break symmetries for efficient optimization in answer set programming. In AAAI, AAAI Press, 6541–6549.

Yang, Z., Ishay, A. and Lee, J. 2020. NeurASP: Embracing neural networks into answer set programming. In IJCAI, ijcai.org, 1755–1762.

Zhang, J., Dridi, M. and El Moudni, A. 2017. A stochastic shortest-path MDP model with dead ends for operating rooms planning. In *ICAC*, IEEE, 1–6.

https://doi.org/10.1017/S1471068425100136 Published online by Cambridge University Press