Inductive Logic Programming for Complications Prediction from Medical Data

Yasmine Akaichi[1[0009−0005−5078−1583]] and
Jean-Marie Jacquet[1[0000−0001−9531−0519]]

Nadi Research Institute, Faculty of Computer Science, University of Namur
Rue Grandgagnage 21, 5000 Namur, Belgium
{yasmine.akaichi,jean-marie.jacquet}@unamur.be

Abstract. There is a significant demand for explainability in machine learning, particularly in critical sectors such as healthcare. In that aim, we use Inductive Logic Programming to automatically learn non-trivial relations between patients' predictive factors of post-intubation complications. We evaluate our system performance based on the interpretability and correctness of the rules as judged by domain experts, as well as through coverage assessments.

Introduction. In the healthcare field, explainability and interpretability are essential for machine learning models. Traditional "black box" models often lack the transparency required for clinical decision-making. In certain medical contexts, such as rare diseases, data is limited, making it difficult to train robust models. Therefore, we require models that can perform efficiently even with small datasets, providing predictions that are based on semantics and logic rather than opaque processes. Moreover, it is crucial to have the ability to incorporate domain knowledge directly into the learning process, allowing the model to reflect expert insights. Additionally, instead of relying on a single model, we need a tool capable of generating multiple models or hypotheses, each explaining the data in different ways, which can lead to deeper understanding and actionable insights.

We propose to use Inductive Logic Programming (ILP) [4, 2], a method that combines machine learning and logic programming, to generate models that are both interpretable and explainable. ILP operates with first-order symbols such as variables, constants, functions, and predicates, which are then used to build more complex structures such as atoms and literals. Given a set of positive and negative examples, along with background knowledge, ILP induces a hypothesis (a set of rules) that generalizes the examples. For this work, we employ Progol [3], a well-known ILP system, which is particularly effective for small datasets, as it can generalize examples regardless of quantity. Importantly, ILP enables the integration of domain knowledge into the learning phase, enriching the model's accuracy and relevance. Furthermore, ILP's output is not restricted to a single model—each induced rule is a standalone model, allowing multiple hypotheses to be explored. We demonstrate the applicability of ILP using a real-world medical case study involving patient data to predict post-intubation complications.

Our Approach. An essential aspect of applying ILP is how data is represented. To train the ILP system, we need structured data in the form of positive and negative examples, supplemented by background knowledge [1]. We preprocessed the dataset to clean special characters and standardize text to lowercase. The dataset consisted of patients' characteristics that serve as predictive factors for post-intubation complications, which is the target class for our predictions. As a result, the complication predicate is the one to learn and is exemplified for each patient as either a positive, as a ground fact of the type `complication(`p_i`)` (complication present for patient p_i) or negative (complication absent) with the negation of such a fact. We enhanced the learning process by incorporating domain knowledge as ground facts or logical clauses. For instance, background information for patient p_i might include conditions such as `short_neck(`p_i`), hta(`p_i`), obese(`p_i`), large_tongue(`p_i`)`. In our case study, domain experts provided key insights into predictive factors, such as the lemon and Cormack scores, and gave us known rules such that a lemon score of 5 or greater indicates a likely difficult intubation, which is expressed as:

```
hard_intubation(P) :- lemonscore(P, score), score ≥ 5.
```

This ability to incorporate expert knowledge into the ILP system allows us to enhance the model's precision and interpretability. The results generated by ILP are both interpretable and explainable. Each rule produced by the ILP system functions as a standalone model, revealing relationships and patterns between patient features and outcomes. For example, in our study, Progol induced several rules. One such rule is:

```
complications(P) :- short_neck(P), hta(P).
```

This rule is straightforward and easy to interpret, clearly linking the presence of a short neck and hypertension (hta) to complications. We also identified several other rules that disclosed patterns between different patient characteristics. Each rule provides a clear explanation of how specific patient features contribute to complications We evaluated the induced rules through both offline and online assessments. Offline, domain experts validated the rules using their medical expertise. Online, we applied the m-estimate to measure rule precision, assessing how well each rule distinguishes between positive and negative cases. The m-estimate smooths the ratio of positive to total examples, providing a balanced and robust evaluation.

Conclusion. This study has demonstrated the effectiveness of ILP in generating interpretable models for predicting post-intubation complications. However, there are still several areas for improvement, which we plan to study by using federated learning and languages including probabilities like ProbLog. On the one hand, while ILP excels with smaller datasets, it faces scalability issues when dealing with larger amounts of data. On the other hand, the healthcare domain frequently contains fuzzy or uncertain data, which presents challenges for traditional ILP systems that assume clean, well-structured inputs.

References

1. Cropper, A., Dumančić, S.: Inductive logic programming at 30: a new introduction. Journal of Artificial Intelligence Research **74**, 765–850 (2022)
2. Cropper, A., Dumancic, S., Muggleton, S.H.: Turning 30: New Ideas in Inductive Logic Programming. IJCAI (2020)
3. Muggleton, S.: Inverse entailment and progol. New generation computing **13**, 245–286 (1995)
4. Muggleton, S., Luc, D.R.: Inductive logic programming: Theory and methods. The Journal of Logic Programming **19**, 629–679 (1994)