Symbolic AI for XAI: Evaluating LFIT Inductive Programming for Fair and Explainable Automatic Recruitment

Alfonso Ortega, Julian Fierrez, Aythami Morales, Zilong Wang
School of Engineering, Universidad Autonoma de Madrid
{alfonso.ortega,julian.fierrez,aythami.morales}@uam.es, zilong.wang@estudiante.uam.es

Tony Ribeiro
Laboratoire des Sciences du Numérique de Nantes National Institute of Informatics Japan
tony_ribeiro@ls2n.fr

Abstract

Machine learning methods are growing in relevance for biometrics and personal information processing in domains such as forensics, e-health, recruitment, and e-learning. In these domains, white-box (human-readable) explanations of systems built on machine learning methods can become crucial. Inductive Logic Programming (ILP) is a subfield of symbolic AI aimed to automatically learn declarative theories about the process of data. Learning from Interpretation Transition (LFIT) is an ILP technique that can learn a propositional logic theory equivalent to a given black-box system (under certain conditions). The present work takes a first step to a general methodology to incorporate accurate declarative explanations to classic machine learning by checking the viability of LFIT in a specific AI application scenario: fair recruitment based on an automatic tool generated with machine learning methods for ranking Curricula Vitae that incorporates soft biometric information (gender and ethnicity). We show the expressiveness of LFIT for this specific problem and propose a scheme that can be applicable to other domains.

1. Introduction

Statistical and optimization-based machine learning algorithms have achieved great success in various applications such as speech recognition [38], image classification [8], machine translation [43], and so on. Among these approaches, deep neural networks have shown the most remarkable success, especially in speech and image recognition. Although deep learning methods usually have good generalization ability on similarly distributed new data, they have some weaknesses including the lack of explanations and the poor understandability by humans of the whole learning process. A deep review about this question can be found in [2].

Another characteristic of these machine learning algorithms is that they rely on data, and therefore reflect those data. This could be an advantage in general, but in some particular domains it could be an important drawback. Consider, for example, automatic recruitment systems, or algorithms to authorize financial products. In these domains, ethic behavior is mandatory and biased data are unacceptable. Appropriate measures have to be taken for guaranteeing ethical AI behavior sometimes contradictory to the possibly biased training data. These questions are receiving increasing interest [1, 9, 25, 39, 40, 20].

On the other hand, logic programming is a declarative programming paradigm with a high level of abstraction. It is based on a formal model (first order logic) to represent human knowledge. Inductive Logic Programming (ILP) has been developed for inductively learning logic programs from examples [22]. Roughly speaking, given a collection of positive and negative examples and background knowledge, ILP systems learn declarative (symbolic) programs [24, 6], which could even be noise tolerant [7, 23], that entail all of the positive examples but none of the negative examples.

One of the ILP most promising approaches for us is Learning From Interpretation Transition (**LFIT**) [30]. **LFIT** learns a logic representation (digital twin) of dynamical complex systems by observing their behavior as a black box under some circumstances. The most general of **LFIT** algorithms is **GULA** (General Usage LFIT Algorithm). **PRIDE** is an approximation to **GULA** with polynomial performance. **GULA** and **PRIDE** generate a propositional logic program equivalent to the system under consideration. These approaches will be introduced in depth in the following sections.

Figure 1 shows the architecture of our proposed ap-

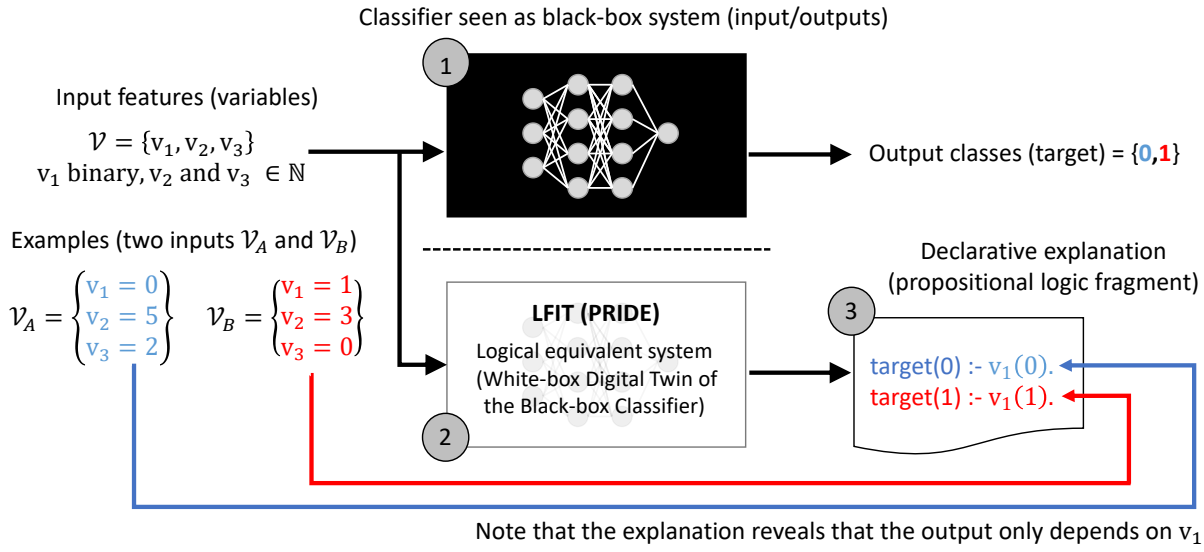

Figure 1: Architecture of the proposed approach for generating an explanation of a given black-box Classifier (1) using **PRIDE** (2) with a toy example (3). Note that the resulting explanations generated by **PRIDE** are in propositional logic.

proach for generating white-box explanations using **PRIDE** of a given black-box classifier.

The main contributions of this work are:

- We have proposed a method to provide declarative explanations using **PRIDE** about the classification process made by an algorithm automatically learnt by a neural architecture in a typical machine learning scenario. Our approach guarantees the logical equivalence between the explanations and the algorithm with respect to the data used to feed **PRIDE**. It does not depend on any particular characteristic of the domain, so it could be applied to any problem.
- We have checked the expressive power of these explanations by experimenting in a multimodal machine learning testbed around automatic recruitment including different biases (by gender and ethnicity).

The rest of the paper is structured as follows: Sec. 2 summarizes the related relevant literature. Sec. 3 describes our methodology including **LFIT**, **GULA**, and **PRIDE**. Sec. 4 presents the experimental framework including the datasets and experiments conducted. Sec. 5 presents our results. Finally Secs. 6, 7 and 8 respectively discuss our work and describe our conclusions and further research lines.

2. Related Works: Inductive Programming for XAI

Some meta-heuristics approaches (as genetic algorithms) have been used to automatically generate programs. Genetic programming (GP) was introduced by Koza [15] for automatically generating LISP expressions for given tasks expressed as pairs input/output. This is, in fact, a typical machine learning scenario. GP was extended by the use of formal grammars to allow to generate programs in any arbitrary language keeping not only syntactic correctness [27] but also semantic properties [28]. Algorithms expressed in any language are declarative versions of the concepts learnt what makes evolutionary automatic programming algorithms machine learners with good explainability.

Declarative programming paradigms (functional, logical) are as old as computer science and are implemented in multiple ways, e.g.: LISP [13], Prolog [5], Datalog [11], Haskell [41], and Answer Set Programs (ASP) [10].

Of particular interest for us within declarative paradigms is logic programming, and in particular first order logic programming, which is based on the Robinson's resolution inference rule that automates the reasoning process of deducing new clauses from a first order theory [17]. Introducing examples and counter examples and combining this scheme with the ability of extending the initial theory with new clauses it is possible to automatically induce a new theory that (logically) entails all of the positive examples but none of the negative examples. The underlying theory from which the new one emerges is considered *background knowledge*. This is the hypothesis of Inductive Logic Programming (ILP, [21, 6]) that has received a great research effort in the last two decades. Recently, these approaches have been extended to make them noise tolerant (in order to overcome one of the main drawbacks of ILP vs statistical/numerical approaches when facing bad-labeled or noisy examples [23]).

Other declarative paradigms are also compatible with ILP, e.g., MagicHaskeller (that implements [14]) with the functional programming language Haskell, and Inductive Learning of Answer Set Programs (ILASP) [16].

It has been previously mentioned that ILP implies some kind of *search* in spaces that can become huge. This search can be eased by hybridising with other techniques, e.g., [26] introduces GA-Progol that applies evolutive techniques.

Within ILP methods we have identified **LFIT** as specially relevant for explainable AI (XAI). In the next section we will describe the fundamentals of **LFIT** and its **PRIDE** implementation, which will be tested experimentally for XAI in the experiments that will follow.

2.1. Learning From Interpretation Transition (LFIT)

Learning From Interpretation Transition (**LFIT**) [12] has been proposed to automatically construct a model of the dynamics of a system from the observation of its state transitions. Given some raw data, like time-series data of gene expression, a discretization of those data in the form of state transitions is assumed. From those state transitions, according to the semantics of the system dynamics, several inference algorithms modeling the system as a logic program have been proposed. The semantics of a system's dynamics can indeed differ with regard to the synchronism of its variables, the determinism of its evolution and the influence of its history.

The **LFIT** framework proposes several modeling and learning algorithms to tackle those different semantics. To date, the following systems have been tackled: memoryless deterministic systems [12], systems with memory [35], probabilistic systems [19] and their multi-valued extensions [36, 18]. The work [37] proposes a method that allows to deal with continuous time series data, the abstraction itself being learned by the algorithm.

In [31, 33], **LFIT** was extended to learn systems dynamics independently of its update semantics. That extension relies on a modeling of discrete memory-less multi-valued systems as logic programs in which each rule represents that a variable possibly takes some value at the next state, extending the formalism introduced in [12, 34]. The representation in [31, 33] is based on annotated logics [4, 3]. Here, each variable corresponds to a domain of discrete values. In a rule, a literal is an atom annotated with one of these values. It allows to represent annotated atoms simply as classical atoms and thus to remain at a propositional level. This modeling allows to characterize optimal programs independently of the update semantics. It allows to model the dynamics of a wide range of discrete systems including our domain of interest in this paper. **LFIT** can be used to learn an equivalent propositional logic program that provides explanation for each given observation.

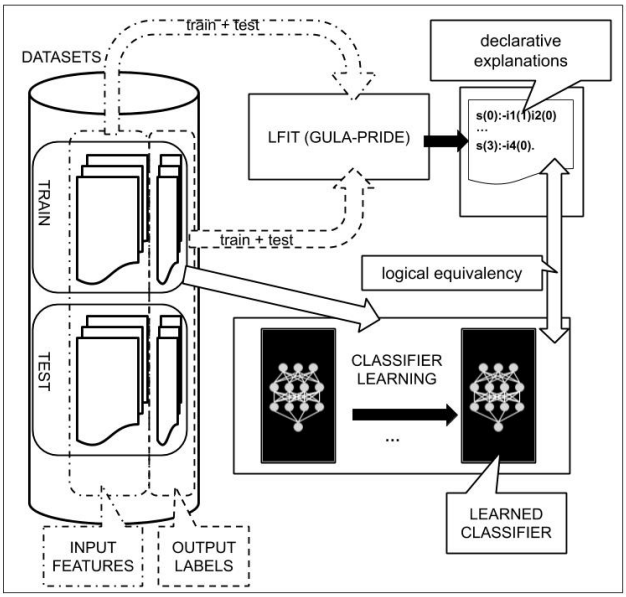

Figure 2: Experimental framework: **PRIDE** is fed with all the data available (train + test) for increasing the accuracy of the equivalence. In our experiments we consider the classifier (see [29] for details) as a black box to perform regression from input resume attributes (atts.) to output labels (recruitment scores labelled by human resources experts). LFIT gets a digital twin to the neural network providing explainability (as human-readable white-box rules) to the neural network classifier.

3. Methods

3.1. General Methodology

Figure 2 graphically describes our proposed approach to generate explanations using **LFIT** of a given black-box classifier. We can see there our purpose to get declarative explanations in parallel (in a kind of white-blox digital twin) to a given neural network classifier. In the present work, for our experiments we have used the same neural network and datasets described in [29] but excluding the face images as it is explained in the following sections.

3.2. PRIDE Implementation of LFIT

GULA [31, 33] and **PRIDE** [32] are particular implementations of the **LFIT** framework [12]. In the present section we introduce notation and describe the fundamentals of both methods.

In the following, we denote by $\mathbb{N} := \{0, 1, 2, ...\}$ the set of natural numbers, and for all $k, n \in \mathbb{N}$, $[\![k; n]\!] := \{i \in \mathbb{N} \mid k \leq i \leq n\}$ is the set of natural numbers between k and n included. For any set S, the cardinal of S is denoted $|S|$ and the power set of S is denoted $\wp(S)$.

Let $\mathcal{V} = \{\mathrm{v}_1, \ldots, \mathrm{v}_n\}$ be a finite set of $n \in \mathbb{N}$ variables, $\mathcal{V}al$ the set in which variables take their values and $\mathsf{dom} : \mathcal{V} \to \wp(\mathcal{V}al)$ a function associating a domain to each variable. The atoms of $\mathcal{M}$VL (multi-valued logic) are of the form v^{val} where $\mathrm{v} \in \mathcal{V}$ and $val \in \mathsf{dom}(\mathrm{v})$. The set of such atoms is denoted by $\mathcal{A}^{\mathcal{V}}_{\mathsf{dom}} = \{\mathrm{v}^{val} \in \mathcal{V} \times \mathcal{V}al \mid val \in \mathsf{dom}(\mathrm{v})\}$ for a given set of variables $\mathcal{V}$ and a given domain function dom. In the following, we work on specific $\mathcal{V}$ and dom that we omit to mention when the context makes no ambiguity, thus simply writing $\mathcal{A}$ for $\mathcal{A}^{\mathcal{V}}_{\mathsf{dom}}$.

Example 1 For a system of 3 variables, the typical set of variables is $\mathcal{V} = \{a, b, c\}$. In general, $\mathcal{V}al = \mathbb{N}$ so that domains are sets of natural integers, for instance: $\mathsf{dom}(a) = \{0, 1\}$, $\mathsf{dom}(b) = \{0, 1, 2\}$ and $\mathsf{dom}(c) = \{0, 1, 2, 3\}$. Thus, the set of all atoms is: $\mathcal{A} = \{a^0, a^1, b^0, b^1, b^2, c^0, c^1, c^2, c^3\}$.

A $\mathcal{M}$VL rule R is defined by:

$$R \;=\; \mathrm{v}_0^{val_0} \leftarrow \mathrm{v}_1^{val_1} \wedge \cdots \wedge \mathrm{v}_m^{val_m} \qquad (1)$$

where $\forall i \in [\![0; m]\!], \mathrm{v}_i^{val_i} \in \mathcal{A}$ are atoms in $\mathcal{M}$VL so that every variable is mentioned at most once in the right-hand part: $\forall j, k \in [\![1; m]\!], j \neq k \Rightarrow \mathrm{v}_j \neq \mathrm{v}_k$. Intuitively, the rule R has the following meaning: the variable v_0 can take the value val_0 in the next dynamical step if for each $i \in [\![1; m]\!]$, variable v_i has value val_i in the current dynamical step.

The atom on the left-hand side of the arrow is called the *head* of R and is denoted $h(R) := \mathrm{v}_0^{val_0}$. The notation $\mathrm{var}(h(R)) := \mathrm{v}_0$ denotes the variable that occurs in $h(R)$. The conjunction on the right-hand side of the arrow is called the *body* of R, written $b(R)$ and can be assimilated to the set $\{\mathrm{v}_1^{val_1}, \ldots, \mathrm{v}_m^{val_m}\}$; we thus use set operations such as $\in$ and $\cap$ on it. The notation $\mathrm{var}(b(R)) := \{\mathrm{v}_1, \cdots, \mathrm{v}_m\}$ denotes the set of variables that occurs in $b(R)$. More generally, for all set of atoms $X \subseteq \mathcal{A}$, we denote $\mathrm{var}(X) := \{\mathrm{v} \in \mathcal{V} \mid \exists val \in \mathsf{dom}(\mathrm{v}), \mathrm{v}^{val} \in X\}$ the set of variables appearing in the atoms of X. A *multi-valued logic program* ($\mathcal{M}$VLP) is a set of $\mathcal{M}$VL rules.

Definition 1 introduces a domination relation between rules that defines a partial anti-symmetric ordering. Rules with the most general bodies dominate the other rules. In practice, these are the rules we are interested in since they cover the most general cases.

Definition 1 (Rule Domination) Let R_1, R_2 be two $\mathcal{M}$VL rules. The rule R_1 *dominates* R_2, written $R_2 \leq R_1$ if $h(R_1) = h(R_2)$ and $b(R_1) \subseteq b(R_2)$.

In [33], the set of variables is divided into two disjoint subsets: $\mathcal{T}$ (for targets) and $\mathcal{F}$ (for features). It allows to define dynamic $\mathcal{M}$VLP which capture the dynamics of the problem we tackle in this paper.

Definition 2 (Dynamic $\mathcal{M}$VLP) Let $\mathcal{T} \subset \mathcal{V}$ and $\mathcal{F} \subset \mathcal{V}$ such that $\mathcal{F} = \mathcal{V} \setminus \mathcal{T}$. A $\mathcal{DM}$VLP P is a $\mathcal{M}$VLP such that $\forall R \in P, \mathrm{var}(h(R)) \in \mathcal{T}$ and $\forall \mathrm{v}^{val} \in b(R), \mathrm{v} \in \mathcal{F}$.

The dynamical system we want to learn the rules of is represented by a succession of *states* as formally given by Definition 3. We also define the "compatibility" of a rule with a state in Definition 4.

Definition 3 (Discrete state) A *discrete state* s on $\mathcal{T}$ (resp. $\mathcal{F}$) of a $\mathcal{DM}$VLP is a function from $\mathcal{T}$ (resp. $\mathcal{F}$) to $\mathbb{N}$, *i.e.* it associates an integer value to each variable in $\mathcal{T}$ (resp. $\mathcal{F}$). It can be equivalently represented by the set of atoms $\{\mathrm{v}^{s(\mathrm{v})} \mid \mathrm{v} \in \mathcal{T}$ (resp. $\mathcal{F})\}$ and thus we can use classical set operations on it. We write $\mathcal{S}^{\mathcal{T}}$ (resp. $\mathcal{S}^{\mathcal{F}}$) to denote the set of all discrete states of $\mathcal{T}$ (resp. $\mathcal{F}$), and a couple of states $(s, s') \in \mathcal{S}^{\mathcal{F}} \times \mathcal{S}^{\mathcal{T}}$ is called a *transition*.

Definition 4 (Rule-state matching) Let $s \in \mathcal{S}^{\mathcal{F}}$. The $\mathcal{M}$VL rule R *matches* s, written $R \sqcap s$, if $b(R) \subseteq s$.

The notion of transition in **LFIT** correspond to a data sample in the problem we tackle in this paper: a couple features/targets. When a rule match a state it can be considered as a possible explanation to the corresponding observation. The final program we want to learn should both:

- match the observations in a complete (all observations are explained) and correct (no spurious explanation) way;
- represent only minimal necessary interactions (according to Occam's razor: no overly-complex bodies of rules)

GULA [31, 33] and **PRIDE** [32] can produce such programs.

Formally, given a set of observations T, **GULA** [31, 33] and **PRIDE** [32] will learn a set of rules P such that all observations are explained: $\forall (s, s') \in T, \forall \mathrm{v}^{val} \in s', \exists R \in P, R \sqcap s, h(R) = \mathrm{v}^{val}$. All rules of P are correct w.r.t. T: $\forall R \in P, \forall (s1, s2) \in T, R \sqcap s1 \implies \exists (s1, s3) \in T, h(R) \in s3$ (if T is deterministic, $s2 = s3$). All rules are minimal w.r.t. $\mathcal{F}$: $\forall R \in P, \forall R' \in \mathcal{M}\mathrm{VLP}, R'$ correct w.r.t. T it holds that $R \leq R' \implies R' = R$.

The possible explanations of an observation are the rules that match the feature state of this observation. The body of the rules gives minimal condition over feature variables to obtain its conclusions over a target variable. Multiple rules can match the same feature state, thus multiple explanations can be possible. Rules can be weighted by the number of observations they match to assert their level of confidence. Output programs of **GULA** and **PRIDE** can also be used in order to predict and explain from unseen feature states by learning additional rules that encode when a target variable value is not possible as shown in the experiments of [33].

4. Experimental Framework

4.1. Dataset

For testing the capability of **PRIDE** to generate explanations in machine learning domains we have designed several experiments using the FairCVdb dataset [29].

FairCVdb comprises 24,000 synthetic resume profiles. Each resume includes 12 features (v_i) related to the candidate merits, 2 demographic attributes (gender and three ethnicity groups), and a face photograph. In our experiments, we discarded the face image for simplicity (unstructured image data will be explored in future work). Each of the profiles includes three target scores (T) generated as a linear combination of the 12 features:

$$T = \beta + \sum_{i=1}^{12} \alpha_i \cdot \mathrm{v}_i, \tag{2}$$

where α_i is a weighting factor for each of the merits (see [29] for details): i) unbiased score ($\beta = 0$); ii) gender-biased scores ($\beta = 0.2$ for male and $\beta = 0$ for female candidates); and iii) ethnicity-biased scores ($\beta = 0.0, 0.15$ and 0.3 for candidates from ethnic groups 1, 2 and 3 respectively). Thus we intentionally introduce bias in the candidate scores. From this point on we will simplify the name of the attributes considering g for gender, e for ethnic group and $i1$ to $i12$ for the rest of input attributes. In addition to the bias previously introduced, some other random bias was introduced relating some attributes and gender to simulate real social dynamics. The attributes concerned were $i3$ and $i7$. Note that merits were generated without bias, assuming an ideal scenario where candidate competencies do not depend on their gender of ethnic group. For the current work we have used only discrete values for each attribute discretizing one attribute (experience to take values from 0 to 5, the higher the better) and the scores (from 0 to 3) that were real valued in [29].

4.2. Experimental Protocol: Towards Declarative Explanations

We have experimented with **PRIDE** on the FairCVdb dataset described in the previous section.

Figure 3 shows names and explains the scenarios considered in our experiments. In [29], researchers demonstrate that an automatic recruitment algorithm based on multimodal machine learning reproduces existing biases in the target functions even if demographic information was not available as input (see [29] for details). Our purpose in the experiments is to obtain a declarative explanation capable of revealing those biases.

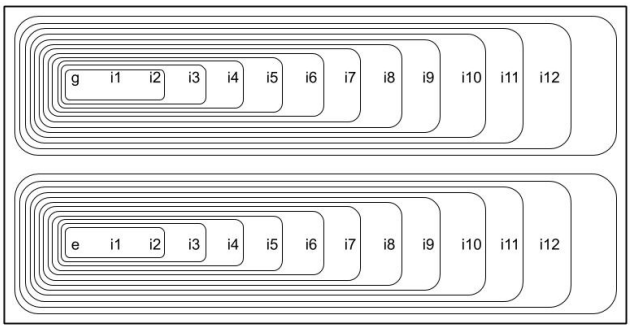

Figure 3: Structure of the experimental tests. There are 4 datasets for analysing gender (named g) and ethnicity (e) bias separately. Apart from gender and ethnicity there are 12 other input attributes (named from $i1$ to $i12$). There is a couple of (biased and unbiased) datasets for each one: gender and ethnicity. We have studied the input attributes by increasing complexity starting with $i1$ and $i2$ and adding one at each time. So, for each couple we have considered 11 different scenarios (named from $s1$ to $s11$). This figure shows their structure (s_i is included in all s_j for which $i < j$).

5. Results

5.1. Example of Declarative Explanation

Listing 1 shows a fragment generated with the proposed methods for scenario $s1$ for gender-biased scores. We have chosen a fragment that fully *explains* how a CV is scored with the value 3 for scenario 1. Scenario 1 takes into account the input attributes gender, education and experience. The first clause (rule), for example, says that if the value of a CV for the attribute gender is 1 (female), for education is 5 (the highest), and for experience is 3, then this CV receives the highest score (3).

The resulting explanation is a propositional logic fragment equivalent to the classifier for the data seen. It can be also understood as a set of rules with the same behavior. From the viewpoint of explainable AI, this resulting fragment can be understood by an expert in the domain and used to generate new knowledge about the scoring of CVs.

```
1
2 scores(3) :- gender(1),
3     education(5),
4     experience(3).
5 scores(3) :- education(4),
6     experience(3).
```

Listing 1: Fragment of explanation for scoring 3

5.2. Quantitative Results: Identifying Biases

Our quantitative results are divided in two parts. The first part is based on the fact that, in the biased experiments, if

$gender(0)$ appears more frequently than $gender(1)$ in the rules, then that would lead to higher scores for $gender(0)$. In the second quantitative experimental part we will show the influence of bias in the distribution of attributes.

5.2.1 Biased attributes in rules

We first define Partial Weight PW as follows. For any program P and two atoms $v_0^{val_0^i}$ and $v_1^{val_1^j}$, where $val_0^i \in val_0$ and $val_1^i \in val_1$, define $S = \forall R \in P \wedge v_0^{val_0^i} \in h(R) \wedge v_1^{val_1^j} \in b(R)$. Then we have: $PW_{v_1^{val_1^j}}(v_0^{val_0^0}) = |S|$. A more accurate PW could be defined, for example, setting different weights for rules with different length. But for our purpose, the frequency is enough. In our analysis, the number of examples for compared scenarios are consistent.

Depending on PW, we define Global Weight GW as follows. For any program P and $v_1^{val_1^j}$, we have: $GW_{v_1^{val_1^j}} = \sum_{val_0^i \in val_0} PW_{v_1^{val_1^j}}(v_0^{val_0^i}) \cdot val_0^i$. The $GW_{v_1^{val_1^j}}$ is a weighted addition of all the values of the output, and the weight, in our case, is the value of scores.

This analysis was performed only on scenario $s11$, comparing unbiased and gender- and ethnicity-biased scores. We have observed a similar behavior of both parameters: Partial and Global Weights. In unbiased scenarios the distributions of the occurrences of each value could be considered statistically the same (between *gender(0)* and *gender(1)* and among *ethnicity(0)*, *ethnicity(1)* and *ethnicity(2)*). Nevertheless in biased datasets the occurrences of *gender(0)* and *ethnic(0)* for higher scores is significantly higher. The maximum difference even triplicates the occurrences of the other values.

For the Global Weights, for example, the maximum differences in the number of occurrences, without and with bias respectively, for higher scores expressed as $\%$ increases from 48.8% to 78.1% for *gender(0)* while for *gender(1)* decreases from 51.2% to 21.9%. In the case of *ethnicity*, it increases from 33.4% to 65.9% for *ethnic(0)*, but decreases from 33.7% to 19.4% for *ethnic(1)* and from 32.9% to 14.7% for *ethnic(2)*.

5.2.2 Distribution of biased attributes

We now define $freq_{p_1}(a)$ as the frequency of attribute a in P_1. The normalized percentage for input a is: $NP_{p_1}(a) = freq_{p_1}(a) / \sum_{x \in input} freq_{p_1}(x)$ and the percentage of the absolute increment for each input from unbiased experiments to its corresponding biased ones is defined as: $AIP_{p_1,p_2}(a) = (freq_{p_1}(a) - freq_{p_2}(a))/freq_{p_2}(a)$.

In this approach we have taken into account all the scenarios (from $s1$ to $s11$) for both *gender* and *ethnicity*.

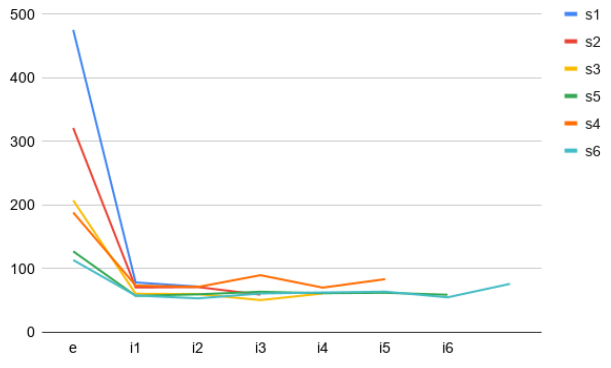

Figure 4: Percentage of the absolute increment (comparing scores with and without bias for ethnicity) of each attribute for scenarios s1, s2, s3, s4, s5 and s6 ($AIP_{us1-6,ebs1-6}$). The graphs link the points corresponding to all the input attributes considered in each scenario.

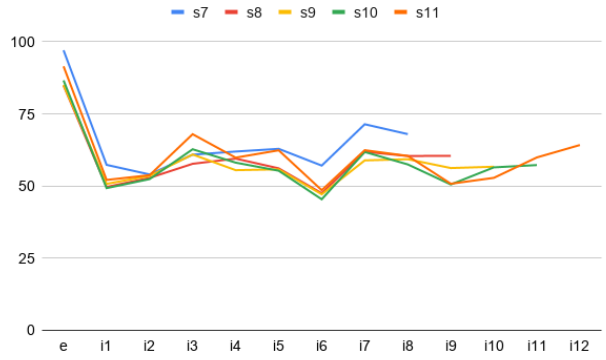

Figure 5: $AIP_{us7-11,ebs7-11}$

We have observed that for both parameters the only attributes that consistently increase their values are *gender* and *ethnicity* comparing unbiased and gender/ethnicity-biased scores. Figures 4 and 5 show $AIP_{us1-11,ebs1-11}$ for each attribute, that is, their values comparing unbiased and ethnic-biased scores for all the scenarios from $s1$ to $s11$. It is clear that the highest values correspond to the attribute *ethnicity*.

Something similar happens for gender. Figures 6 and 7 show $AIP_{us1-11,gbs1-11}$ for each attribute when studying gender-biased scores. It is worth mentioning some differences in scenarios $s9$, $s10$ and $s11$ regarding attributes $i3$ and $i7$. These apparent anomalies are explained by the random bias introduced in the datasets in order to relate these attributes with gender when the score is biased. Figure 8 shows NP_{s11} for all the attributes. It clearly shows the small relevance of attributes $i3$ and $i7$ in the final bi-

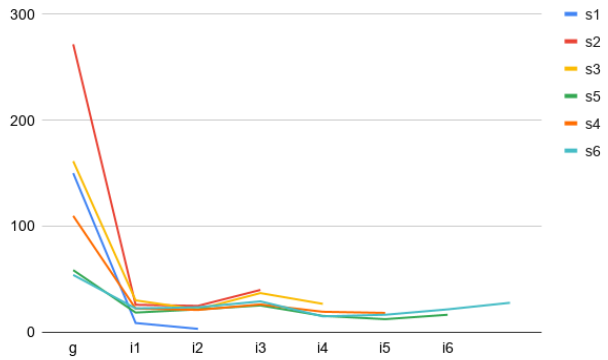

Figure 6: $AIP_{us1-6,ebs1-6}$

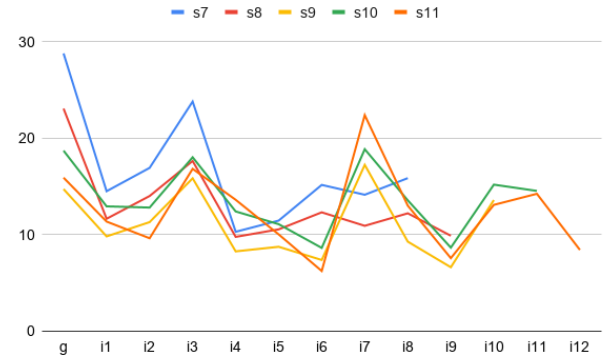

Figure 7: $AIP_{us7-11,gbs7-11}$

ased score. As it is highlighted elsewhere, this capability of **PRIDE** to identify this random indirect perturbation of other attributes in the bias is a relevant achievement of our proposal.

6. Discussion

After running the experiments described in the previous sections we can extract the following conclusions.

- **PRIDE can *explain* algorithms learnt by neural networks.** The theorems that support the characteristics of PRIDE allow *to get a set of propositional clauses logically equivalent to the systems* observed when facing the input data provided. In addition, *each proposition has a set of conditions that is minimum*. So, once the scorer is learnt, PRIDE translates it into a logical equivalent program. This program is a list of clauses like the one shown in Listing 1. Logical programs are declarative theories that explain the knowledge on a domain.

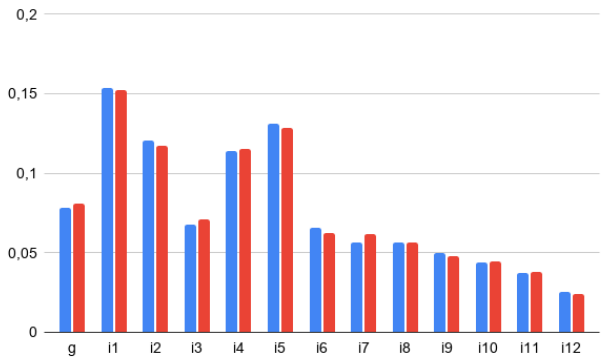

Figure 8: Normalized percentage of frequency in scenario s11 of each attribute: g, i1 to i11 (NP_{s11}). No bias (blue), Gender-biased scores (red).

- **PRIDE can *explain* what happens in a specific domain.** Our experimental results discover these characteristics of the domain:
  - ***Insights into the structure of the datasets***. We have seen (and further confirmed with the authors of the datasets) some characteristics of the datasets, e.g.: 1) *All the attributes are needed for the score*. We have learnt the logical version of the system starting from only two input attributes and including one additional attribute at a time and we only reached an accuracy of 100 % when taking into account all of them. This is because removing some attributes generates indistinguishable CVs (all the remainder attributes have the same value) with different scores (that correspond to different values in some of the removed attributes). 2) *Gender and ethnicity are not the most relevant attributes for scoring*: The number of occurrences of these attributes is much smaller than others in the conditions of the clauses of the learnt logical program. 3) While trying to catch the biases we have discovered that *some attributes seem to increase their relevance when the score is biased*. For example, the competence in some specific languages (attribute i7) seems to be more relevant when the score has gender bias. After discussing with the authors of the datasets, they confirmed a random perturbation of these languages into the biases, that explained our observations.
  - ***Biases in the training datasets are detected.*** We have analysed the relationship between the scores and the specific values of the attributes used to generated the biased data. We have proposed

a simple mathematical model based on the *effective weights* of the attributes that concludes that higher values of the scores correspond to the same specific values of gender (for gender bias) and ethnic group (for ethnicity bias). On the other hand, we have performed an exhaustive series of experiments to analyse the increase of the presence of the gender and ethnicity in the conditions of the clauses of the learnt logical program (comparing the unbiased and biased versions).

Our overall conclusion is that **LFIT**, and in particular **PRIDE**, is able to offer explanations to the algorithm learnt in the domain under consideration. The resulting explanation is, as well, expressive enough to catch training biases in the models learnt with neural networks.

7. Conclusions

The main goal of this paper was to check if ILP (and more specifically LFIT with **PRIDE**) could be useful to provide *declarative explanations* in machine learning by neural networks.

The domain selected for our experiments in this first entry to the topic is one in which the explanations of the learned models' outputs are specially relevant: automatic recruitment algorithms. In this domain, ethic behavior is needed, no spurious biases are allowed. For this purpose, a pack of synthetically generated datasets has been used. The datasets contain resumes (CVs) used in [29] for testing the ability of deep learning approaches to reproduce and remove biases present in the training datasets. In the present work, different input attributes (including the resume owner merits, gender, and ethnicity) are used to score each CV automatically using a neural network. Different setups are considered to introduce artificial gender- and ethnicity-bias in the learning process of the neural network. In [29] face images were also used and the relationship between these pictures and the biases was studied (it seems clear that from the face you should be able to deduce the gender and ethnic group of a person). Here we have removed images because PRIDE is more efficient with pure discrete information.

Our main goal indicated above translates into these two questions: Is PRIDE expressive enough to explain how the program learnt by deep-learning approaches works? Does PRIDE catch biases in the deep-learning processes? We have given positive answer to both questions.

8. Further Research Lines

- **Increasing understandability.** Two possibilities could be considered in the future: 1) to *ad hoc* post-process the learnt program for translating it into a more abstract form, or 2) to increase the expressive power of the formal model that supports the learning engine using, for example, ILP based on first order logic.
- **Adding predictive capability. PRIDE** is actually not aimed to predict but to explain (declaratively) by means of a digital twin of the observed systems. Nevertheless, it is not really complicated to extend **PRIDE** functionality to predict. It should be necessary to change the way in which the result is interpreted as a logical program: mainly by adding mechanisms to chose the most promising rule when more than one is applicable.

  Our plan is to test an extended-to-predict **PRIDE** version to this same domain and compare the result with the classifier generated by deep learning algorithms.
- **Handling numerical inputs.** [29] included as input the images of the faces of the owners of the CVs. Although some variants to **PRIDE** are able to cope with numerical signals, the huge amount of information associated with images implies performance problems. Images are a typical input format in real deep learning domains. We would like to add some automatic pre-processing step for extracting discrete information (such as semantic labels) from input images. We are motivated by the success of systems with similar approaches but different structure like [42].
- **Measuring the accuracy and performance of the explanations.** As far as the authors know there is no standard procedure to evaluate and compare different explainability approaches. We will incorporate in future versions some formal metric.

9. Acknowledgements

This work has been supported by projects: PRIMA (H2020-MSCA-ITN-2019-860315), TRESPASS-ETN (H2020-MSCA-ITN-2019-860813), IDEA-FAST (IMI2-2018-15-853981), BIBECA (RTI2018-101248-B-I00 MINECO/FEDER), RTI2018-095232-B-C22 MINECO, and Accenture.

References

[1] A. Acien, A. Morales, R. Vera-Rodriguez, I. Bartolome, and J. Fierrez. Measuring the gender and ethnicity bias in deep models for face recognition. In *Iberoamerican Congress on Pattern Recognition (IBPRIA)*, pages 584–593. Springer, 2018.

[2] Alejandro Barredo Arrieta, Natalia Díaz Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020.

[3] H. A Blair and V.S. Subrahmanian. Paraconsistent foundations for logic programming. *Journal of Non-classical Logic*, 5(2):45–73, 1988.

[4] H. A. Blair and V.S. Subrahmanian. Paraconsistent logic programming. *Theoretical Computer Science*, 68(2):135 – 154, 1989.

[5] I. Bratko. *Prolog Programming for Artificial Intelligence, 4th Edition*. Addison-Wesley, 2012.

[6] A. Cropper and S. H. Muggleton. Learning efficient logic programs. *Machine Learning*, 108(7):1063–1083, 2019.

[7] W. Z. Dai, S. H. Muggleton, and Z. H. Zhou. Logical vision: Meta-interpretive learning for simple geometrical concepts. 2015.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

[9] P. Drozdowski, C. Rathgeb, A. Dantcheva, N. Damer, and C. Busch. Demographic bias in biometrics: A survey on an emerging challenge. *IEEE Transactions on Technology and Society*, 2020.

[10] M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub. *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2012.

[11] S. S. Huang, T. Jeffrey Green, and B. T. Loo. Datalog and emerging applications: an interactive tutorial. In Timos K. Sellis, Renée J. Miller, Anastasios Kementsietsidis, and Yannis Velegrakis, editors, *Proc. of the ACM SIGMOD Intl. Conf. on Management of Data*, pages 1213–1216, June 2011.

[12] K. Inoue, T. Ribeiro, and C. Sakama. Learning from interpretation transition. *Machine Learning*, 94(1):51–79, 2014.

[13] Guy L. Steele Jr. *Common LISP: The Language, 2nd Edition*. Digital Pr., 1990.

[14] S. Katayama. Systematic search for lambda expressions. In M. C. J. D. van Eekelen, editor, *Revised Selected Papers from the Sixth Symposium on Trends in Functional Programming*, volume 6 of *Trends in Functional Programming*, pages 111–126. Intellect, September 2005.

[15] J.R. Koza. *Genetic Programming*. MIT Press, 1992.

[16] M. Law. Inductive learning of answer set programs., 2018. PhD. Imperial College London.

[17] J. W. Lloyd. *Foundations of Logic Programming, 2nd Edition*. Springer, 1987.

[18] D. Martınez, G. Alenya, C. Torras, T. Ribeiro, and K. Inoue. Learning relational dynamics of stochastic domains for planning. In *Proceedings of the 26th International Conference on Automated Planning and Scheduling*, 2016.

[19] D. Martínez Martínez, T. Ribeiro, K. Inoue, G. Alenyà Ribas, and C. Torras. Learning probabilistic action models from interpretation transitions. In *Proceedings of the Technical Communications of the 31st International Conference on Logic Programming (ICLP 2015)*, pages 1–14, 2015.

[20] Aythami Morales, Julian Fierrez, Ruben Vera-Rodriguez, and Ruben Tolosana. Sensitivenets: Learning agnostic representations with application to face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[21] S. Muggleton. Inductive logic programming. In S. Arikawa, Sh. Goto, S. Ohsuga, and T. Yokomori, editors, *Proc. First Intl. Workshop on Algorithmic Learning Theory*, pages 42–62. Springer, October 1990.

[22] S. Muggleton. Inductive logic programming. *New Generation Computing*, 8(4):295–318, 1991.

[23] S. Muggleton, W.-Z. Dai, C. Sammut, A. Tamaddoni-Nezhad, J. Wen, and Z.-H. Zhou. Meta-interpretive learning from noisy images. *Machine Learning*, 107(7):1097–1118, 2018.

[24] S. H. Muggleton, D. Lin, N. Pahlavi, and A. Tamaddoni-Nezhad. Meta-interpretive learning: application to grammatical inference. *Machine Learning*, 94(1):25–49, 1994.

[25] S. Nagpal, M. Singh, R. Singh, M. Vatsa, and N.i K. Ratha. Deep learning for face recognition: Pride or prejudiced? *CoRR*, abs/1904.01219, 2019.

[26] A. T. Nezhad. Logic-based machine learning using a bounded hypothesis space: the lattice structure, refinement operators and a genetic algorithm approach, August 2013. PhD, Imperial College London.

[27] M. O'Neill and R. Conor. *Grammatical Evolution - Evolutionary Automatic Programming in an Arbitrary Language*, volume 4 of *Genetic programming*. Kluwer, 2003.

[28] A. Ortega, M. de la Cruz, and M. Alfonseca. Christiansen grammar evolution: Grammatical evolution with semantics. *IEEE Trans. Evol. Comput.*, 11(1):77–90, 2007.

[29] A. Peña, I. Serna, A. Morales, and J. Fierrez. Bias in multimodal AI: testbed for fair automatic recruitment. In *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 129–137, June 2020.

[30] T. Ribeiro. Studies on learning dynamics of systems from state transitions, 2015. PhD.

[31] T. Ribeiro, M. Folschette, M. Magnin, O. Roux, and K. Inoue. Learning dynamics with synchronous, asynchronous and general semantics. In *International Conference on Inductive Logic Programming*, pages 118–140. Springer, 2018.

[32] T. Ribeiro, M. Folschette, L. Trilling, N. Glade, K. Inoue, M. Magnin, and O. Roux. Les enjeux de l'inférence de modèles dynamiques des systèmes biologiques à partir de séries temporelles. In C. Lhoussaine and E. Remy, editors, *Approches symboliques de la modélisation et de l'analyse des systèmes biologiques*. ISTE Editions, 2020. In edition.

[33] T. Ribeiro, M. Folschette, M.and Magnin, and K. Inoue. Learning any semantics for dynamical systems represented by logic programs. working paper or preprint, Sept. 2020.

[34] T. Ribeiro and K. Inoue. Learning prime implicant conditions from interpretation transition. In *Inductive Logic Programming*, pages 108–125. Springer, 2015.

[35] T. Ribeiro, M. Magnin, K. Inoue, and C. Sakama. Learning delayed influences of biological systems. *Frontiers in Bioengineering and Biotechnology*, 2:81, 2015.

[36] T. Ribeiro, M. Magnin, K. Inoue, and C. Sakama. Learning multi-valued biological models with delayed influence from time-series observations. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 25–31, Dec 2015.

[37] T. Ribeiro, S. Tourret, M. Folschette, M. Magnin, D. Borzacchiello, F. Chinesta, O. Roux, and K. Inoue. Inductive learning from state transitions over continuous domains. In N. Lachiche and C. Vrain, editors, *Inductive Logic Programming*, pages 124–139. Springer, 2018.

[38] A. Senior, V. Vanhoucke, P. Nguyen, and T. Sainath. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing magazine*, 2012.

[39] Ignacio Serna, Aythami Morales, Julian Fierrez, Manuel Cebrian, Nick Obradovich, and Iyad Rahwan. Algorithmic discrimination: Formulation and exploration in deep learning-based face biometrics. In *AAAI Workshop on Artificial Intelligence Safety (SafeAI)*, February 2020.

[40] Ignacio Serna, Alejandro Peña, Aythami Morales, and Julian Fierrez. InsideBias: Measuring bias in deep networks and application to face gender biometrics. In *IAPR Intl. Conf. on Pattern Recognition (ICPR)*, January 2021.

[41] S. J. Thompson. *Haskell - The Craft of Functional Programming, 3rd Edition*. Addison-Wesley, 2011.

[42] D. Varghese and A. Tamaddoni-Nezhad. One-shot rule learning for challenging character recognition. In S. Moschoyiannis, P. Fodor, J. Vanthienen, D. Inclezan, Ni. Nikolov, F. Martín-Recuerda, and I. Toma, editors, *Proc. of the 14th Intl. Rule Challenge*, volume 2644 of *CEUR Workshop Proceedings*, pages 10–27, June 2020.

[43] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, and J Klingner. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.