Explainable AI for Sperm Morphology: Integrating YOLO with FastLAS*

Talissa Dreossi[1,3,*], Agostino Dovier[1,3], Susy Urli[2], Francesca Corte Pause[2] and Martina Crociati[2]

[1]Dip. di Scienze Matematiche, Informatiche e Fisiche, Università degli Studi di Udine, 33100 Udine, Italy
[2]Dip. di Scienze Agroalimentari, Ambientali e Animali, Università degli Studi di Udine, 33100 Udine, Italy
[3]GNCS-INdAM, Gruppo Nazionale per il Calcolo Scientifico.

Abstract

Deep learning models excel in complex classification tasks but often lack interpretability, limiting their adoption in domains where explainability is critical, such as medicine and veterinary science. This work presents a hybrid approach that combines deep learning and symbolic reasoning to classify bull spermatozoa morphology in an explainable manner. We utilise YOLOv8 for object detection and morphological and viability classification of bull's spermatozoa from microscope-acquired images, achieving high accuracy. To tackle explainability, FastLAS was employed to learn human-readable classification rules. These rules, coupled with the xASP2 framework, enable traceable justifications for each classification, addressing the black-box nature of deep learning. Experimental evaluation demonstrates that, while FastLAS does not match YOLO's performance, it outperforms traditional machine learning models and offers significant benefits in explainability. This approach provides a practical solution for integrating explainable AI in reproductive biology, with implications for medical AI systems where transparency is essential.

Keywords

Inductive Logic Programming, ASP, AI Explainability, Object Detection

1. Introduction

Artificial Intelligence (AI) systems have been widely adopted, including also fields where it is essential to know the reasoning behind their outcomes. An evident example is the medical domain, where explainability is crucial for ensuring trust and informed decision-making. Indeed, while deep learning models achieve remarkable accuracy in predictive tasks, their black-box nature raises concerns in domains where decision transparency is mandatory. One possible solution is to use built-in interpretable approaches, such as *Logic Programming* (LP). However, these systems often struggle to achieve the predictive performance required for complex classification tasks. To address this, we propose a hybrid approach combining both techniques.

The case discussed in this study is bull's sperm morphology analysis, a fundamental step to discriminate between satisfactory and unsatisfactory bulls intended as potential breeders. Traditional evaluation methods rely on visual assessment of semen smears performed by trained operators, requiring significant time and expertise. Moreover, inter-operator variability can lead to inconsistencies in classification, affecting the reliability of the analysis. The automatization of this process with AI could improve efficiency and standardization, but the opacity of *Neural Networks* (NN) remains a limiting factor. To address this, we combined deep learning with symbolic reasoning. Specifically, we use the *You Only Look*

CILC 2025: 40th Italian Conference on Computational Logic, June 25-27, 2025, Alghero, Italy

*The work is partially supported by Interdepartmental Project on AI (Strategic Plan UniUD–22-25) and by PNRR FAIR MAPSART - CUP C63C22000770006 and by Unione europea-Next Generation EU, missione 4 componente 2, project MaPSART "Future Artificial Intelligence (FAIR)"

*Corresponding author.
talissa.dreossi@uniud.it (T. Dreossi); agostino.dovier@uniud.it (A. Dovier); susy.urli@uniud.it (S. Urli); francesca.cortepause@uniud.it (F. Corte Pause); martina.crociati@uniud.it (M. Crociati)
https://dmif.uniud.it/dovier (A. Dovier)
0009-0007-1746-6500 (T. Dreossi); 0000-0003-2052-8593 (A. Dovier); 0009-0009-4360-2929 (S. Urli); 0000-0001-7073-7575 (F. Corte Pause); 0000-0003-1651-7795 (M. Crociati)

Once (YOLO) object detection algorithm to identify and classify spermatozoa from microscope-acquired images. To favour interpretability, we then employ the *Inductive Logic Programming* (ILP) system FastLAS [1] to learn classification rules from YOLO's predictions, thus generating an ASP program that predicts or confirm predictions, and finally xASP for explaining the decisions made with an easy-to-read output. We have combined all these modules in an app, called *XAI-BSC* (*eXplainable AI for Bull Sperm Classification*), which, given an image, detect the spermatozoa automatically using YOLO and then gives an explanation in natural language using ASP rules. Even if the system we are presenting is not yet completely automatic, requiring sometimes human intervention to extract features for the part of explainability, practical experiments with domain experts prove that it is already a useful tool that allows saving time to operators.

2. Related Works.

Bringing advanced computational tools into medical and veterinary fields has greatly improved how accurately we can make diagnoses and how efficiently things get done. Automated software systems, such as Computer-Aided Sperm Analysis (CASA), have been successfully implemented in order to assess sperm viability and motility in fresh semen samples [2]. Furthermore, with AI it was possible to develop predictive statistical models that can assist in decision-making processes across a wide range of fields, including medical and veterinary applications [3, 4]. For instance, AI algorithms are able to accurately predict the time of calving in dairy cows, aiding farmers in managing livestock more efficiently [5, 6]. In a similar way, deep learning models have also been applied in the detection of lung abnormalities in cats [7].

Recent progress in object detection techniques, especially with algorithms like YOLO [8, 9], have shown significant promise in medical imaging. Indeed, studies like the ones by Chen et al. [10] and Prinzi et al. [11] have highlighted the ability of YOLO in detecting, respectively, cell and cancer.

Inductive Logic Programming has also made notable progress. In particular we will consider the development of FastLAS [1]. FastLAS offers significant potential for creating interpretable and accurate rule-based models. It has proven to perform exceptionally well when more specific rules are preferred over general ones. For example Drozdov et al. [12] used FastLAS to build a security policy system, where restrictive rules were prioritized. Similarly, in some of our previous works we employed FastLAS for weather forecasting, to explain complex models in an interpretable manner [13], and in legal reasoning, to extract and possibly explain judges' reasoning [14]. Additionally, FastLAS has also demonstrated its effectiveness in combination with NN [15]. By integrating NN with symbolic reasoning, this approach aims to combine the learning capabilities of NN with the interpretability of symbolic methods. A smilar work that integrate those techniques was made by Collevati et. al [16].

Regarding logic programming and explainability, xASP2 has emerged as a significant advancement [17]. xASP2 focuses on improving the understandability of *Answer Set Programming* (ASP) by providing a framework to generate explanations for the solutions produced by ASP. Specifically, xASP2 has the ability to link the presence or absence of an atom in an answer set to the logic rules involved in its inference. Other systems, such as xclingo [18], DiscASP [19], xASP [20], exp(ASPc) [21] could also be used to tackle this task. xASP2 currently handles better certain important aspects, such as explaining false atoms or supporting specific advanced language features.

The paper is organized as follows: in Sect. 3 we give some backgrounds on the problem, on logic programming and machine learning languages and tools used in this work. A brief overview of the related work on deep learning approaches to the problem faced in this paper is reported in Sect. 4 while the Inductive Logic Programming approach is subject of Sect. 5. Sect. 6 describes experiments, while Conclusions and some possible lines for future work are drawn in Sect. 7.

3. Background

Understanding the morphological features of bull spermatozoa is essential for assessing fertility. The analysis is traditionally performed through visual evaluation of spermatozoa under bright-field microscopy following the *Society For Theriogenology* (SFT) as part of the *Bull Breeding Soundness Evaluation* (BBSE). In standard practice, sperm morphology evaluation is performed independently by at least two technicians. The agreement between their assessments typically ranges from 43% (Fair) to 58% (Moderate), ensuring that the analysis remains reliable.

In this study, we employed two techniques to achieve this evaluation: YOLO, a deep learning-based object detection model, and FastLAS, an ILP framework designed for learning interpretable rules from data, since it can handle numerical variables. While it would have been possible to define classification rules with the help of experts, we opted to use an ILP framework for two main reasons. First, while expert guidelines exist, many morphological features (e.g., the degree of redness required to classify a sperm cell as dead) cannot be captured with fixed thresholds. As a result, these thresholds cannot be arbitrarily encoded and must instead be learned from data. Second, inter-operator variability in manual classification introduces inconsistencies, which can be mitigated through a data-driven approach. By using FastLAS, we aim to infer classification rules directly from data, thereby enhancing consistency and objectivity.

3.1. YOLO

YOLOv8 is a one-stage object detection model that can localize and classify objects in a single pass, which is more efficient than two-stage models like Faster R-CNN. YOLOv8 processes images by passing them through a *Convolutional Neural Network* (CNN), which extracts spatial and semantic features at multiple levels. YOLO does not scan the image with sliding windows, neither generates region proposals, but indeed it divides the image into an $S \times S$ grid, where S is a tunable hyperparameter. Each grid cell is responsible for predicting bounding boxes and class probabilities within its region of the image. In order to determine the bounding boxes, YOLOv8 employs an anchor-free detection. This helps in speeding up the computation since an anchor-based approach uses predefined boxes which can slow learning, especially when the model is applied to custom datasets.

The model architecture is composed by three main components: *backbone*, *neck*, and *head*. The backbone is a pre-trained CNN and its purpose is to extract hierarchical feature maps. Then, there is the neck which combines these features using a *Feature Pyramid Network* (FPN). FPN allows to improve the model's ability to detect objects of varying sizes. Finally, the head computes an output which consist of: (i) a bounding box coordinates (x, y, w, h), where (x, y) denote the centre and (w, h) the dimensions, (ii) a confidence score indicating object presence and bounding box accuracy, and (iii) class probabilities for object classification.

3.2. Answer Set Programming

ASP is a declarative programming paradigm, which is usually employed to model and solve problems that involve non-monotonic reasoning. An ASP program consists of a set of rules that encode properties and constraints of a given problem. Each rule has the general form: H `:-` $A_1, \ldots, A_n, \texttt{not}\, B_1, \ldots, \texttt{not}\, B_m$ where H, A_i, and B_j are *atoms*, each expressed as $p(t_1, \ldots, t_k)$, where p is a predicate symbol, and t_i are terms (constants or variables). H denotes the *head* of the rule, representing the conclusion inferred when the *body* (the other part of the rule) is satisfied. The symbol `:-` denotes implication from body to head. A rule is *ground* if it contains no variables. ASP supports two forms of literals: atoms A and their default negation $\texttt{not}\, A$, called *naf-literal* (negation as failure). Default negation enables the expression of incomplete information and supports non-monotonic reasoning, where conclusions may be withdrawn in light of new evidence.

ASP semantics is the one of *stable models* (also called *answer sets*): a set of ground atoms S is a stable model of a program P if it is the unique minimal model of the reduct P^S, derived by eliminating all rules whose bodies are not satisfied by S, and by removing all default negation from the bodies of the remaining

rules [22]. A key property of stable models is that for any atom $A \in S$, there exists at least one rule r in the ground version of P such that the body of r is satisfied by S and A is the head of r. This provides a basis for explanation: not only is A known to be true in S, but it is also possible to identify the supporting rule(s) and the literals that justify its derivation.

3.3. Inductive Logic Programming

Inductive Logic Programming is a subfield of machine learning that focuses on learning logical rules from *examples* and *background knowledge*. The learning process involves searching rules within an *hypothesis space*, to identify those that best explain the given examples while being also consistent with the background knowledge.

Among the various frameworks available in ILP, we adopted *Learning from Answer Sets* (LAS) as it exploits ASP and is therefore particularly well-suited for learning non-monotonic logical programs. A learning problem in LAS is formalised as a task, represented by the tuple $T = \langle B, S_M, E \rangle$, where: B denotes the background knowledge, which is a set of ASP rules, S_M is the hypothesis space, i.e. the set of rules to be possibly learned, and E represents the set of examples that the knowledge base extended with that learned rules must explain.

Instead of a complete listing of the hypothesis space, usually it is implicitly defined by a mode bias, specified as a pair $\langle M_h, M_b \rangle$, where M_h and M_b are sets of mode declarations for the heads and bodies of rules, respectively. Each mode declaration describes which predicates and argument types are allowed in the corresponding part of a rule. Arguments in a mode declaration can be either `var(t)` or `const(t)`, indicating a variable or constant of type `t`. A literal is said to be compatible with a given mode declaration if it can be formed by substituting each `var(t)` with a variable of type t, and each `const(t)` with a constant of that type.

Examples are represented as *partial interpretations*. Formally, a partial interpretation e_{pi} is a pair of sets of ground atoms $\langle e^{inc}, e^{exc} \rangle$, where e^{inc} denotes the inclusions, which are atoms that must be true, and e^{exc} denotes the exclusions, the atoms that must be false when inclusions are true. A candidate interpretation I is said to extend a partial interpretation e_{pi} if it satisfies the conditions $e^{inc} \subseteq I$ and $e^{exc} \cap I = \emptyset$. Additionally, examples in LAS can be context-dependent, meaning their interpretation must be evaluated relative to specific contextual information. A *context-dependent partial interpretation* (CDPI) is defined as a tuple $e_{cdpi} = \langle e_{pi}, e_{ctx} \rangle$, where e_{pi} is a partial interpretation and e_{ctx} is an ASP program representing the context in which the example should be interpreted. A program P is said to accept a CDPI e_{cdpi} if there exists at least one stable model A of $P \cup e_{ctx}$ such that A extends e_{pi}. This allows LAS to handle diverse examples under varying conditions, making it well-suited for tasks requiring context-sensitive reasoning.

A strong point of LAS, implemented by the system FastLAS [1], is its ability to handle noisy data, a crucial feature when working with real-world datasets, which often contain inconsistencies or errors. LAS models noise by associating a penalty to each example, allowing some of them to be violated during learning, if doing so leads to a better overall hypothesis. If they are not covered, then the hypothesis will have a higher cost, as we will better describe later. To represent noisy examples, LAS uses *Weighted Context-Dependent Partial Interpretations* (WCDPIs). A WCDPI is defined as a tuple $e = \langle e_{id}, e_{pen}, e_{cdpi} \rangle$, where e_{id} is a unique identifier for the example, e_{pen} is a positive integer indicating the penalty incurred if the example is not covered by the hypothesis, and e_{cdpi} is a context-dependent partial interpretation as described earlier. A LAS task that incorporates noisy examples is called a *Noisy LAS task*, formally defined as the tuple $T^{noise} = \langle B, S_M, E \rangle$, where B is an ASP program providing background knowledge, S_M is the hypothesis space, and E is a finite set of WCDPIs. A hypothesis $H \subseteq S_M$ is an inductive solution of T^{noise} if and only if $B \cup H$ accepts every example $e \in E$.

We said that if a hypothesis fails to accept a particular example, it *pays the penalty* of that example. In fact, this penalty contributes to the overall cost of the hypothesis, representing the trade-off between hypothesis's explanatory power and its failure to cover certain examples. The cost function of a hypothesis is indeed defined as the sum over the penalties of all of the examples that are not accepted by the hypothesis, augmented with the length of the hypothesis. For this reason, the objective of noisy LAS tasks is to

identify an *optimal solution*, i.e. an hypothesis that minimises the cost function within a given hypothesis space, based on a set of WCDPI examples.

3.4. ASP Explainability

xASP2 [17] is a tool developed for explaining computed answer sets. Precisely, given a program Π, an answer set S, and an atom α, xASP2 aims to answer queries of the form "Why does $\alpha \in S$ (resp., $\alpha \notin S$)" by identifying the subset of Π that supports this inclusion (resp., exclusion). It generates explanations in the form of justification graphs, specifically *Directed Acyclic Graphs* (DAGs), where nodes represent atoms and edges represent rules. Edges from node A to node B indicate that the atom in B appears in the body of the rule on the edge, while the atom in A appears in the head of the rule. For each atom in a stable model, xASP2 identifies a supporting rule whose body is true and only contains atoms that have already been explained, ensuring the acyclicity of the graph. Two main challenges in providing such explanations are: (i) how to compute a small set of assumptions capable of explaining the assignment of $\alpha \in A$, and (ii) how to handle sophisticated constructs like choice rules and aggregates, which may be involved in explaining the falsity of certain atoms in an easily understandable way. While logic programs under the answer set semantics can also produce explanations based on the assumption of false atoms, this approach would often result in vague explanations for all false atoms, potentially reducing the explanation to mere assumptions.

To begin, xASP2 requires a preprocessing step to obtain a justification for a ground program P. The computation starts with a three-valued interpretation, denoted as (I^+, I^-), where both sets are initially empty. A three-valued interpretation consists of a pair (L,U), where L and U are sets of ground atoms, with $L \subseteq U$ and neither set contains the atom $\bot$. The aim is to iteratively expand these sets to determine which atoms are true or false based on the rules of the program.

The I^+ set is expanded through iterative inference steps. In the three-valued setting, a rule's head can only be inferred if every atom b in its body is determined, that is, $b \in I^+$ or $b \in I^-$. The I^- set, on the other hand, is expanded by iteratively adding atoms that belong to an unfounded set $X \subseteq B_P$, where B_P is the set of ground atoms of rules' body in P. An unfounded set X is defined with respect to a ground logic program P if, for each atom $a \in X$, and for each rule $r \in P$ whose head is a, one of the following conditions must hold: either one of the literals in the body of the rule is known to be false, meaning that if b appears positively in the body, it must be in I^-, and if b appears negatively, it must be in I^+; or for some atom b appearing positively in r, it must hold that $b \in X$. The purpose of identifying and adding atoms to the I^- set is to eliminate "self-referential" sets of atoms. These are sets where each atom in the set is supported by another atom within the set, but none can be inferred to be true by other means. Such sets can lead to cyclic justifications that do not provide meaningful explanations. Removing them, xASP2 ensures that only non-cyclic, valid justifications are kept.

After preprocessing, xASP2 generates an explanation graph that is similar to the one used in *offline justification* [23]. The graph visualizes the relationships between the atoms in the program based on the inference process. Throughout this process, xASP2 tracks the rules used at each step of the derivation and labels the edges of the explanation graph accordingly.

Finally, xASP2 uses a metaprogramming approach with the Clingo Python API and provides a flexible Python interface. Users can request explanations for individual literals or opt to generate multiple graphs covering all atoms in a given answer set. It also offers an interactive, web-based environment navigator for visualising and exploring these explanation graphs.

4. Deep Learning Module

We provide here a very brief overview of the deep learning module [24, 25], as the primary focus of this work is on the ILP module and the explainability framework.

4.1. Dataset and Methods

The initial dataset contained 4,890 microscopy fields with 1 to 20 eosin-nigrosine-stained spermatozoa, labelled by experts. Data augmentation (brightness, contrast, blur, and noise) expanded the dataset to 8,243 images, annotated with bounding boxes using labelImg [26]. Spermatozoa were classified according to SFT guidelines into six categories based on morphology and viability: *Normal Alive* (NA) and *Normal Dead* (ND) spermatozoa, *Major Abnormalities Alive* (MAA) and *Major Abnormalities Dead* (MAD) spermatozoa, *minor Abnormalities Alive* (mAA), and *minor Abnormalities Dead* (mAD) spermatozoa.

Training was initially set to 100 epochs with early stopping to prevent overfitting. If performance improved beyond 100 epochs, training was extended by 50 epochs. The optimal model, often achieved around 100 epochs, was saved based on validation loss, even if reached earlier. As an initial work, the YOLO network was not customized in any way.

4.2. Results

The model [25] achieved 82% mean accuracy, though performance varied due to class imbalance, especially among under-represented anomaly classes. Prediction quality was assessed using the *Intersection over Union* (IoU) metric:

$$IoU = \frac{GT \cap Pr}{GT \cup Pr} \tag{1}$$

where GT and Pr are the areas of ground-truth and predicted boxes, respectively, and so $GT \cap Pr$ represents the area of overlap, while $GT \cup Pr$ represents the area of union. We also report *mean Average Precision* (mAP): mAP50 (IoU ≥ 0.50) reached 79%, and mAP50-95 (IoU 0.50–0.95) was 44%, reflecting detection and localization performance.

All considered, the obtained accuracy value of 82% can be considered satisfactory because in a real-world scenario, an 80-85% coefficient of agreement between two operators is considered “moderate to very strong" [27].

5. ILP Module

We explore the FastLAS encoding of the dataset and its results. The background knowledge just includes predicates for the domain of the features considered.

5.1. Examples

To encode the examples used in this study, a custom-developed application, that we will call *XAI-BSC*, was employed. XAI-BSC was designed to facilitate the process of extracting key morphological features from the spermatozoa images, which were then used to generate the necessary data for learning with FastLAS. XAI-BSC has also the explainability features that we will describe later. Since the ultimate goal of this study is to use LP to make DL models explainable, no machine learning or other black-box techniques were used in the development of the ILP module. The retrieval of morphological information was achieved through traditional computer vision techniques.

As a first step, XAI-BSC takes the manually annotated bounding boxes, i.e. the YOLO annotation file, and the corresponding image as input. The app then displays one spermatozoon at a time and asks the user to answer a series of questions regarding its morphological characteristics. These questions help assess key features that we are not able yet to automatically detect, such as the way the tail is curled, although they can sometimes be inferred indirectly from other features (e.g., the aspect ratio of the bounding box). Next, the app attempts to outline the spermatozoon’s head contour: this contour provides valuable geometric information, often essential for classifying different sperm classes; for instance, if the shape resembles a pear, it is indicative of major anomalies.

To detect and accurately outline the heads of spermatozoa in microscopic images, we follow a two-step process: first, we detect candidate head regions using the *Hough Circle Transform* (HCT); then, we extract

precise contours using *active contour models*. The first step identifies potential sperm heads based on their bright, approximately circular appearance. The HCT operates by first detecting edges in the image, typically using the *Canny* edge detector. For each edge point, the algorithm votes in an accumulator space for all possible circles that could pass through it. These votes represent degrees of radial symmetry, with higher symmetry corresponding to a stronger likelihood of a circle at that location. Peaks in this accumulator space indicate the most probable circle centres and radii, which are then selected as candidate head regions.

Once potential head regions are identified, we restrict this set by filtering the size and then we refine the detection by accurately tracing the contour of each head. This is achieved using active contour models, commonly known as *snakes*. The method evolves an initial contour towards the true boundary based on image gradients and smoothness constraints. For each detected circle, a snake is initialized as a larger circle ($3\times$ radius) around the detected centre. This provides a starting point for the snake to converge onto the actual boundary. The snake is then evolved over a *Gaussian-smoothed* image, which reduces noise while preserving important features. The result is a smooth, continuous contour that accurately follows the boundary of the sperm head. However, due to the nature of the images, which are almost grayscale with low contrast, detecting the contour can sometimes be challenging. Additionally, since multiple sperm heads may be present in a single image, it is not easy to automatically establish which of the detected contours corresponds to the target of interest.

Figure 1: Examples of contour detection of the head. Correct: green/left, Wrong: (right/any of them)

When the contour is not identifiable (see Fig. 1 right), the app defaults to using only the data derived from the bounding box: those measurement comprehends redness, ratio and area. The image's redness level serves as a key indicator for viability assessment: if the spermatozoon is red then it is classified as dead, otherwise is alive. However, detecting the sperm head alone is not always possible, we measure redness across the entire image instead. The app also computes the ratio of the bounding box, which provides insight into whether the spermatozoon has a tail, is curled, or exhibits other morphological irregularities. Additionally, the area of the bounding box is calculated, offering further information about the size of the sperm.

On the other hand, when the contour can be detected (see Fig. 1 left), the app extracts additional features. It calculates the length of the contour, the area enclosed by the contour (which provides a precise measurement of the sperm head's size), and the ratio of the minimum and maximum axes of the contour. These features contribute to a more detailed understanding of the sperm's shape. The roundness of the sperm head is also determined using the formula:

$$Roundness = \frac{4 \times \pi \times A}{p^2} \tag{2}$$

where A is the area and p is the perimeter. This measure quantifies the circularity of the head, helping to distinguish normal sperm from those exhibiting abnormalities, as mentioned earlier.

The detection of the tail was even more challenging as its shape can vary significantly across different anomalies. However, by calculating the aspect ratio of the bounding box around the spermatozoon, we

gain some insight into tail morphology: the closer this ratio is to one, the more likely it is that the tail is curled, coiled or missing.

Two separate training sessions were conducted in this study. The first one focused on training to distinguish between alive and dead spermatozoa, while the second training aimed to differentiate between anomalies: no anomalies (`"n"`), major anomalies (`"M"`), and minor anomalies (`"m"`). This approach facilitated training by allowing the ILP module to focus on learning one feature at a time, unlike the DL module, which must infer all characteristics simultaneously. Below we report an encoding example. Their general form is `#pos(id@w, {inc}, {exc}, {cntx})`, where `#pos` indicate that it is a positive example (while for negatives it is `#neg`), `id` is an unique identifier for the example, `w` is the weight of the example (i.e. the measure of representativeness of an example), and then we have the three sets of ground atoms of WCDPI examples, also explained in Section 3. `inc` represents the inclusions of the example, *i.e.* atoms that must be true, while `exc`, the exclusions, are atoms that cannot be true when inclusions are. Finally, `cntx` is the context in which the example must hold.

```
#pos(id58_12@100,
    {label("M")},                                    % inclusions
    {label("n"), label("m"), curledTail,             % exclusions
    missingTail, singleCoilTail, overtunedHead},
    {red(648). area(25200). ratio(175).              % context
    headRoundness(74). headArea(2594). lenghtHead(210).
    ratioHead(2). bubbleHead.}).
```

As the reader can observe, the inclusion set contains the class label, which, in this case, regards anomalies. However, the same procedure applies to viability classification. In the exclusions, we include all other classes (meaning that, it can belong only to one class of anomaly at a time) and any morphologies that were not selected by the user within the app. The context of the data contains all the other characteristics retrieved either automatically from the image or manually through XAI-BSC.

In this study, all examples were assigned equal weight (100), as the number of examples for each class was balanced, and we did not have any specific insight into which examples might be more or less representative, or which could be considered noisy, for each class. If such information had been available, we could have adjusted the weights accordingly, assigning higher weights to the more representative examples and lower weights to the noisy ones.

To improve model's generalizability and address scalability limitations of ILP, cross-validation with 10 folds was employed during training. This approach allowed us to include a wider variety of examples, reducing the overfitting risk and ensuring robustness. Each fold of the training dataset contained 12 spermatozoa per class, with 7 per class in the test sets.

5.2. Hypothesis Space and Bias

FastLAS employs a penalty function to guide the rule learning by assigning a cost to the rules components, ensuring that the most relevant features are considered.

For the training that distinguishes between alive and dead spermatozoa, a strong bias is introduced to encourage rules that generalize across multiple examples. Indeed, we assigned a penalty of 100 to the head of the rule and 1 to each body predicate: covering a single example thus entails a minimum cost of 101. This configuration forces rules to cover multiple examples simultaneously, as failing to cover a single example already carries a significant cost of 100 (see Section 5.1). However, other similar biases could also be effective.

For the training that distinguishes between normal spermatozoa and those with major or minor anomalies, a different penalty scheme is used. Compared to the viability classification task, the penalty assigned to the head of a rule is lower (10 instead of 100), reflecting the fact that anomalies can be determined by a wider range of morphological features, making rules more complex. Regarding the predicates in the body we can categorize them in two groups. The first one includes predicates about features that are specific of a class of anomalies (e.g., missing or curled tail), therefore it has the lowest

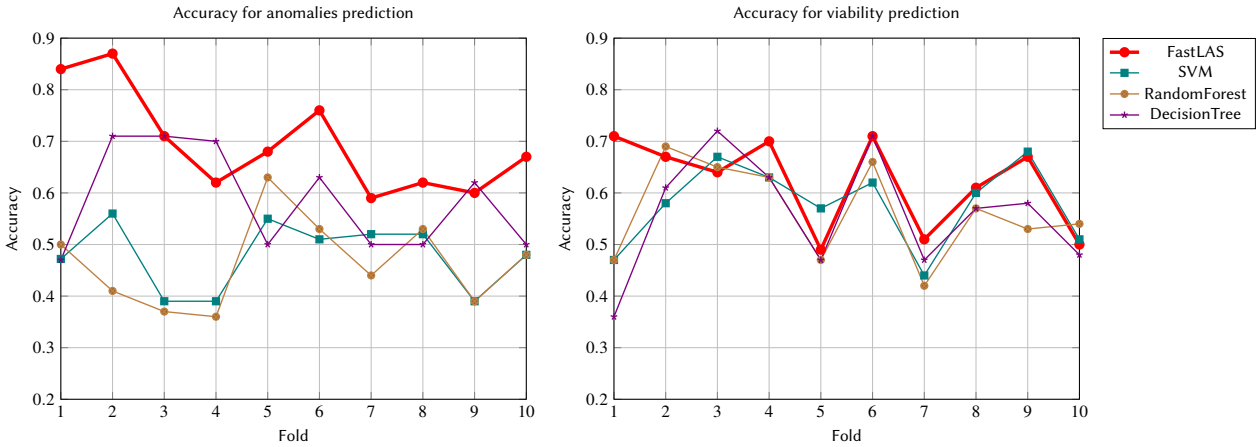

Figure 2: Accuracy for each fold for each model. On the left, the graph relative to prediction of anomalies and, on the right, the one relative to viability

cost (3) since these are strong indicators of anomalies. For example, a spermatozoon missing its tail is highly likely to have minor anomalies, making these features sufficient for classification. The second one has a slightly higher cost and include all the other features. Specifically, redness has a penalty of 5 since it is more relevant for viability classification than for detecting anomalies. Similarly, head length is penalized more because it is not as informative as other morphological features. Meanwhile, head area, ratio, and roundness have intermediate penalties (4), as they aid classification but are less decisive than the first group.

Overall, while the penalty differences between features are relatively small, they help balance rule complexity and generalizability, ensuring that the learned rules prioritize the most discriminative characteristics while avoiding overfitting.

5.3. Results

Given the inherently different nature of deep learning and logic-based approaches, it is unsurprising that FastLAS exhibits significantly lower performance in prediction compared to YOLO. However, this was expected, as the primary goal of using FastLAS is not to outperform deep learning but to provide explainability in decision-making. The rules by FastLAS are similar in structure to the following:

```
pred_label("m") :- distalDroplet, ratio(V_0_rr), V_0_rr >= 77.
pred_label("alive") :- red(V_0_r), V_0_r <= 386.
```

which comply with expert knowledge. Anyway, to evaluate the effectiveness of the ILP approach, we compared FastLAS against traditional machine learning models such as *Support Vector Machines* (SVM), *Decision Trees*, and *Random Forests*, which have been proven to be actually able to learn from data, even if not as much as a DL system. As shown in the plots (Fig. 2), FastLAS consistently outperforms these models across various metrics, demonstrating its ability to extract meaningful rules from the data.

The Table 1 summarises the performance of the models in classifying spermatozoa into normal, major or minor anomaly categories. The metrics are divided into two macro-averaging (M) and micro-averaging (μ) as explained below.

$$Precision_M = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathrm{TP}_i}{\mathrm{TP}_i+\mathrm{FP}_i} \qquad Precision_\mu = \frac{\sum_{i=1}^{N}\mathrm{TP}_i}{\sum_{i=1}^{N}(\mathrm{TP}_i+\mathrm{FP}_i)}$$

$$Recall_M = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathrm{TP}_i}{\mathrm{TP}_i+\mathrm{FN}_i} \qquad Recall_\mu = \frac{\sum_{i=1}^{N}\mathrm{TP}_i}{\sum_{i=1}^{N}(\mathrm{TP}_i+\mathrm{FN}_i)}$$

where TP, FP and FN are respectively true positive, false positive and false negative. FastLAS achieves 70% accuracy and a weighted F1-score of 0.73, indicating a reasonable precision-recall balance. The higher recall (0.79) in micro-averaged results suggests effective anomaly detection, though there may be a trade-off in precision.

For the classification of spermatozoa as either alive or dead, we refer to Table 2. With 62% an accuracy and a weighted F1-score of 0.63, FastLAS shows reasonable performance in distinguishing alive from dead spermatozoa. The recall of 0.65 indicates that the system correctly identifies a fair proportion of the positive cases, although there is room for improvement.

Table 1
Performance comparison of different models for anomalies prediction.

	Precision_M	Recall_M	Precision_μ	Recall_μ	Accuracy	Weighted F1
FastLAS	0.70	0.69	0.78	0.79	0.70	0.73
SVM	0.33	0.48	0.44	0.48	0.48	0.40
Random Forest	0.46	0.46	0.45	0.46	0.46	0.46
Decision Tree	0.47	0.45	0.46	0.45	0.58	0.45

Table 2
Performance comparison of different models for viability prediction.

	Precision	Recall	Accuracy	Weighted F1
FastLAS	0.63	0.65	0.62	0.63
SVM	0.73	0.54	0.58	0.55
Random Forest	0.64	0.54	0.48	0.40
Decision Tree	0.52	0.55	0.56	0.56

Overall, while the performance of FastLAS is lower than that of YOLO, it remains competitive with traditional machine learning models and offers the added advantage of explainability, which is a key objective of this study. Furthermore we want to underline that we will be using YOLO for the prediction (with an accuracy above 80%), and rules from FastLAS for explainability. We are not comparing YOLO results here since it was trained on a wider set of images, with a different data preprocessing.

6. Explainability

As said the aim of this work is developing a system able to explain the prediction of the DL module. The ASP rules we obtained from FastLAS are indeed extremely interpretable, but we still need to reconstruct the reasoning process, i.e. the rules that are triggered to get the correct prediction. For this purpose, we used xASP2.

6.1. System Description

We developed XAI-BSC[1], a python-based application (python version 3.10) that integrates YOLO with ASP. The workflow begins with the user uploading an image, which is processed by YOLO to detect and classify the spermatozoa. The model's predictions are saved, and the image, overlaid with bounding boxes and predicted labels, is presented to the user, who can select a specific spermatozoon for which they wish to obtain an explanation.

Once selected, the chosen spermatozoon is displayed in isolation, using the sub-image defined by the predicted bounding box. As in the dataset preparation for FastLAS, we collect some information about the sperm's shape directly from the user, while other morphological features are automatically extracted by XAI-BSC. These features are then encoded into logical facts in ASP format, following the same schema used during rule learning with FastLAS.

[1]code available at https://clp.dimi.uniud.it/sw/

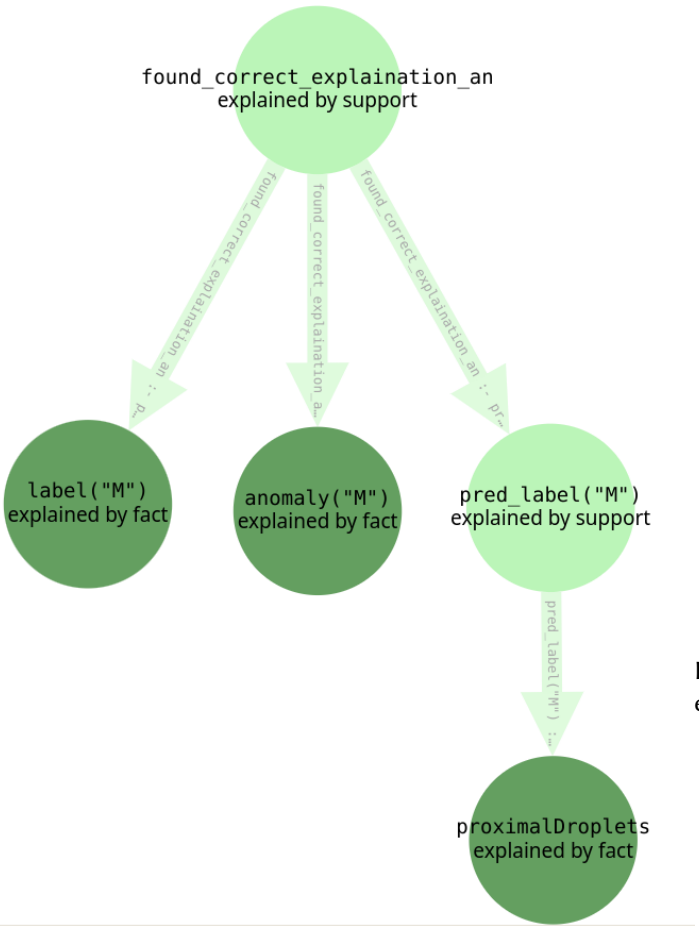

Figure 3: xASP2 DAG for the anomaly prediction in Fig. 4.

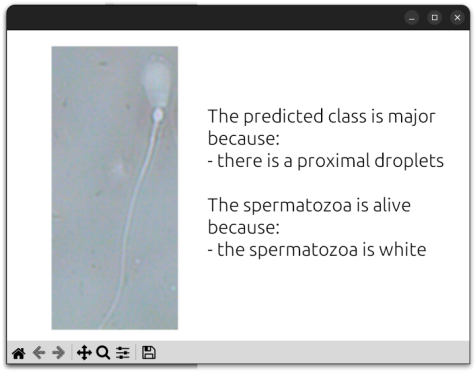

Figure 4: Image of spermatozoa with explanation of anomaly and viability.

The feature encoding and the rules learned by FastLAS are provided as input to clingo, which computes the answer set corresponding to the classification process. Using xASP2, we generate a graph representing this answer set, enabling us to visualise the reasoning steps that led to the final prediction. From this graph, we trace the path of inference that supports the classification. It is important to note that the ASP rules may occasionally result in no prediction or incorrect predictions. Therefore, not all classifications are explainable within the symbolic framework. To ensure that explanations correspond to the YOLO prediction, we include an auxiliary predicate (`found_correct_explanation`) that holds true only when the ASP-derived label matches YOLO's output. The other ASP prediction are ignored. Once the correct reasoning path is identified, it is translated into natural language, providing the user with a clear and understandable explanation for the classification decision. Specifically, from the DAG generated by xASP2, the algorithm searches for all links starting from the correct explanation node. From the prediction node, it follows links to other feature or fact nodes, extracts the variable values from these nodes for later substitution into explanations.

XAI-BSC and tests (see Section 5.3) were conducted on a system running Ubuntu 24.04 LTS with an Intel 12th Gen Core i7-12700 processor and a maximum clock speed of 4.9 GHz with 16 GB RAM. It featured an NVIDIA GeForce RTX 3060 GPU and an SK Hynix NVMe SSD.

6.2. Results

Figs. 3, 4 and 5 illustrate the system's explanation capabilities for class predictions. The first two figures presents the explanation for a sample predicted as having a major anomaly. In this case, the system highlights the presence of a proximal droplet as the key feature justifying the classification. This information is shown both in natural language alongside the image of the spermatozoon and in the form

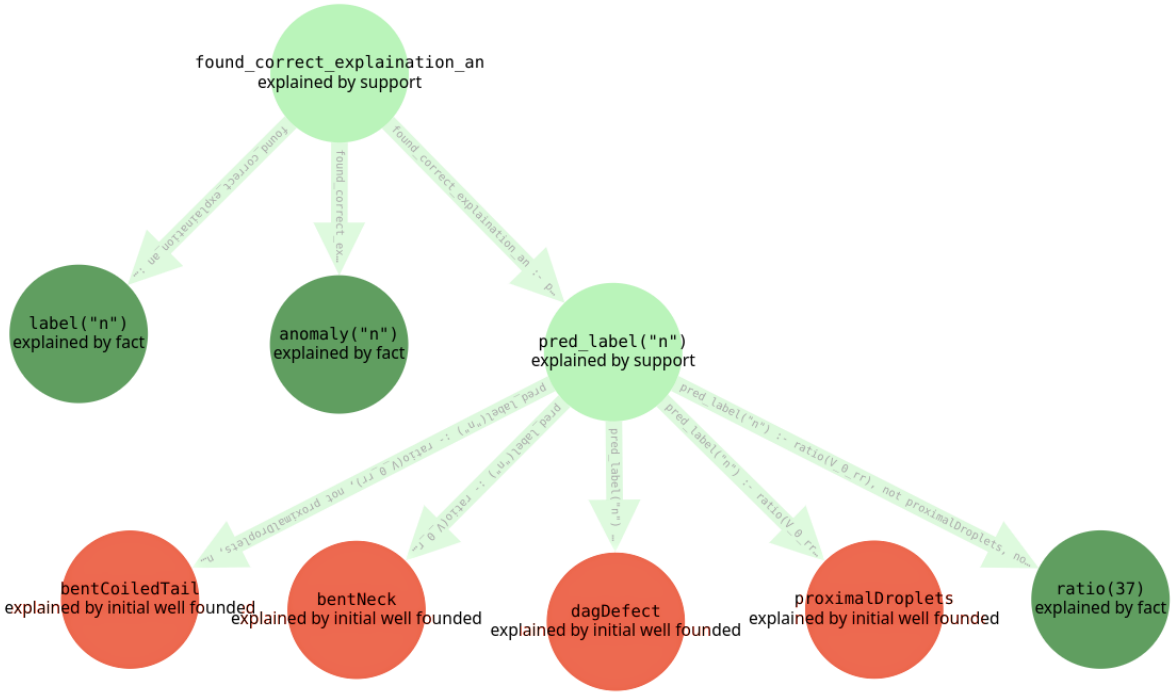

Figure 5: xASP2 DAG for a prediction of no anomalies (normal)

of an explanation graph. On the other hand, Fig. 5 shows the explanation for a sample classified as having no anomalies. Unlike the previous case, the classification here is supported by the absence (red nodes) of specific features such as bent tail, bent neck, or proximal droplets. Additionally, the system takes into account the ratio value of 0.37 (scaled for ASP encoding), which further supports the prediction of a normal spermatozoon.

7. Conclusions and Future Works

By integrating YOLO, for object detection, with FastLAS, for rule learning, we have presented XAI-BSC, which has been proved to be a tool able to achieve both accurate classification and interpretable explanations.

We encountered some limitations using FastLAS, such as for scalability when the number of features, examples or hypothesis space grow significantly. Our strategy to mitigate this was to restrict the learning to separate sub-tasks (viability and anomaly classification). Instead, to improve generalizability, we adopted a 10-fold cross-validation strategy which ensured that rules are exposed to a wider variety of examples and so are not tailored to a single dataset split. This also permitted to guarantee robustness since rules are learnt from a broad set of different examples.

Looking ahead, we are planning several improvements and extensions. First, enhancing the accuracy of sperm head detection (even if we can already detect heads with a decent accuracy). Additionally, we aim to develop reliable methods for tail detection and automate the identification of other morphological features such as proximal droplets. These improvements will help reduce manual input and increase the system's autonomy. Finally, we plan to conduct user studies to evaluate the system's explainability in practice, assessing how effectively it supports decision-making and user trust in real-world settings, and we will also compare our work with other explainable AI methods such as LIME or SHAP.

Acknowledgments

The authors would like to acknowledge the valuable contributions of Prof. Alessandro Dal Palù for his support in the elaboration of image with computer vision techniques and Prof. Giuseppe Stradaioli for his assistance in the early stages of the YOLO-based application.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] M. Law, A. Russo, E. Bertino, K. Broda, J. Lobo, FastLAS: Scalable inductive logic programming incorporating domain-specific optimisation criteria, 2020. URL: http://dx.doi.org/10.1609/aaai.v34i03.5678. doi:10.1609/aaai.v34i03.5678.

[2] C. O'Meara, E. Henrotte, K. Kupisiewicz, C. Latour, M. Broekhuijse, A. Camus, L. Gavin-Plagne, E. Sellem, The effect of adjusting settings within a computer-assisted sperm analysis (CASA) system on bovine sperm motility and morphology results, Animal Reproduction 19 (2022). URL: http://dx.doi.org/10.1590/1984-3143-AR2021-0077. doi:10.1590/1984-3143-ar2021-0077.

[3] M. Shehab, L. Abualigah, Q. Shambour, M. A. Abu-Hashem, M. K. Y. Shambour, A. I. Alsalibi, A. H. Gandomi, Machine learning in medical applications: A review of state-of-the-art methods, Computers in Biology and Medicine 145 (2022) 105458. URL: http://dx.doi.org/10.1016/J.COMPBIOMED.2022.105458. doi:10.1016/j.compbiomed.2022.105458.

[4] R. B. Appleby, P. S. Basran, Artificial intelligence in veterinary medicine, Journal of the American Veterinary Medical Association 260 (2022) 819–824. URL: http://dx.doi.org/10.2460/JAVMA.22.03.0093. doi:10.2460/javma.22.03.0093.

[5] M. Crociati, L. Sylla, A. De Vincenzi, G. Stradaioli, M. Monaci, How to predict parturition in cattle? a literature review of automatic devices and technologies for remote monitoring and calving prediction, Animals 12 (2022) 405. URL: http://dx.doi.org/10.3390/ani12030405. doi:10.3390/ani12030405.

[6] M. Borchers, Y. Chang, K. Proudfoot, B. Wadsworth, A. Stone, J. Bewley, Machine-learning-based calving prediction from activity, lying, and ruminating behaviors in dairy cattle, Journal of Dairy Science 100 (2017) 5664–5674. URL: http://dx.doi.org/10.3168/jds.2016-11526. doi:10.3168/jds.2016-11526.

[7] L. Dumortier, F. Guépin, M.-L. Delignette-Muller, C. Boulocher, T. Grenier, Deep learning in veterinary medicine, an approach based on CNN to detect pulmonary abnormalities from lateral thoracic radiographs in cats, Scientific Reports 12 (2022). URL: http://dx.doi.org/10.1038/s41598-022-14993-2. doi:10.1038/s41598-022-14993-2.

[8] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You Only Look Once: Unified, real-time object detection, 2015. URL: https://arxiv.org/abs/1506.02640. doi:10.48550/ARXIV.1506.02640.

[9] M. Hussain, YOLO-v1 to YOLO-v8, the rise of YOLO and its complementary nature toward digital manufacturing and industrial defect detection, Machines 11 (2023) 677. URL: http://dx.doi.org/10.3390/machines11070677. doi:10.3390/machines11070677.

[10] X. Chen, H. Zheng, H. Tang, F. Li, Multi-scale perceptual YOLO for automatic detection of clue cells and trichomonas in fluorescence microscopic images, Computers in Biology and Medicine 175 (2024) 108500. URL: http://dx.doi.org/10.1016/J.COMPBIOMED.2024.108500. doi:10.1016/j.compbiomed.2024.108500.

[11] F. Prinzi, M. Insalaco, A. Orlando, S. Gaglio, S. Vitabile, A YOLO-based model for breast cancer detection in mammograms, Cognitive Computation 16 (2023) 107–120. URL: http://dx.doi.org/10.1007/S12559-023-10189-6. doi:10.1007/s12559-023-10189-6.

[12] A. Drozdov, M. Law, J. Lobo, A. Russo, M. W. Don, Online symbolic learning of policies for

explainable security, in: 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), IEEE, Atlanta, GA, USA, 2021, p. 269–278. URL: http://dx.doi.org/10.1109/TPSISA52974.2021.00030. doi:10.1109/tpsisa52974.2021.00030.

[13] T. Dreossi, A. Dovier, A. Formisano, M. Law, A. Manzato, A. Russo, M. Tait, Towards explainable weather forecasting through FastLAS, in: Logic Programming and Nonmonotonic Reasoning, Springer Nature Switzerland, Dallas, Texas, 2024, p. 262–275. URL: http://dx.doi.org/10.1007/978-3-031-74209-5_20. doi:10.1007/978-3-031-74209-5_20.

[14] A. Dovier, T. Dreossi, A. Formisano, XAI-LAW towards a logic programming tool for taking and explaining legal decisions, in: Proceedings of the 39th Italian Conference on Computational Logic, 2024, volume 3733 of *CEUR Workshop Proceedings*, CEUR-WS.org, Rome, Italy, 2024. URL: https://ceur-ws.org/Vol-3733/short3.pdf.

[15] D. Cunnington, A. Russo, M. Law, J. Lobo, L. Kaplan, NSL: Hybrid interpretable learning from noisy raw data, 2020. URL: https://arxiv.org/abs/2012.05023. doi:10.48550/ARXIV.2012.05023.

[16] M. Collevati, T. Eiter, N. Higuera, Leveraging neurosymbolic ai for slice discovery, in: Neural-Symbolic Learning and Reasoning, Springer Nature Switzerland, Cham, 2024, pp. 403–418.

[17] M. Alviano, L. L. Trieu, T. Cao Son, M. Balduccini, Explanations for answer set programming, Electronic Proceedings in Theoretical Computer Science 385 (2023) 27–40. URL: http://dx.doi.org/10.4204/EPTCS.385.4. doi:10.4204/eptcs.385.4.

[18] P. Cabalar, J. Fandinno, B. Muñiz, A system for explainable answer set programming, Electronic Proceedings in Theoretical Computer Science 325 (2020) 124–136. URL: http://dx.doi.org/10.4204/EPTCS.325.19. doi:10.4204/eptcs.325.19.

[19] F. Li, H. Wang, K. Basu, E. Salazar, G. Gupta, DiscASP: A graph-based ASP system for finding relevant consistent concepts with applications to conversational socialbots, Electronic Proceedings in Theoretical Computer Science 345 (2021) 205–218. URL: http://dx.doi.org/10.4204/EPTCS.345.35. doi:10.4204/eptcs.345.35.

[20] L. L. Trieu, T. C. Son, M. Balduccini, xASP: An explanation generation system for answer set programming, in: Logic Programming and Nonmonotonic Reasoning, Springer International Publishing, Genova, Italy, 2022, p. 363–369. URL: http://dx.doi.org/10.1007/978-3-031-15707-3_28. doi:10.1007/978-3-031-15707-3_28.

[21] L. L. Trieu, T. C. Son, M. Balduccini, exp(ASPc): Explaining asp programs with choice atoms and constraint rules, Electronic Proceedings in Theoretical Computer Science 345 (2021) 155–161. URL: http://dx.doi.org/10.4204/EPTCS.345.28. doi:10.4204/eptcs.345.28.

[22] M. Gelfond, V. Lifschitz, The stable model semantics for logic programming, in: R. Kowalski, Bowen, Kenneth (Eds.), Proceedings of International Logic Programming Conference and Symposium, MIT Press, Seattle, Washington, USA, 1988, pp. 1070–1080. URL: http://www.cs.utexas.edu/users/ai-lab?gel88.

[23] E. Pontelli, T. C. Son, Justifications for Logic Programs Under Answer Set Semantics, Springer Berlin Heidelberg, 2006, p. 196–210. URL: http://dx.doi.org/10.1007/11799573_16. doi:10.1007/11799573_16.

[24] S. Urli, F. Corte Pause, T. Dreossi, G. Stradaioli, Implementation of an artificial intelligence system for morphological evaluation of spermatozoa in routine bull semen analysis, 2024. Poster presented at EAAP conference, Firenze, 2024.

[25] S. Urli, F. C. Pause, T. Dreossi, M. Crociati, G. Stradaioli, Evaluation of an artificial intelligence system for bull sperm morphology evaluation, Theriogenology (2025) 117504.

[26] D. Tzutalin, labelImg, https://github.com/tzutalin/labelImg, 2015.

[27] R. Hanson, S. Reddick, S. Thuerauf, K. Webb, V. Kasimanickam, R. Kasimanickam, Comparison of bull sperm morphology evaluation methods under field conditions, Clinical Theriogenology 15 (2023). URL: http://dx.doi.org/10.58292/CT.V15.9425. doi:10.58292/ct.v15.9425.