



HAL
open science

Knowledge-driven active learning

Gabriele Ciravegna, Frédéric Precioso, Marco Gori

► **To cite this version:**

Gabriele Ciravegna, Frédéric Precioso, Marco Gori. Knowledge-driven active learning. 2022. hal-03526510

HAL Id: hal-03526510

<https://hal.science/hal-03526510>

Preprint submitted on 14 Jan 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

KNOWLEDGE-DRIVEN ACTIVE LEARNING

Gabriele Ciravegna

Università di Firenze¹
Università Côte d’Azur²
¹Italy, ²France

Frederic Precioso

Université Côte d’Azur
France

Marco Gori

Università di Siena¹
Università Côte d’Azur²
¹Italy, ²France

ABSTRACT

In the last few years, Deep Learning models have become increasingly popular. However, their deployment is still precluded in those contexts where the amount of supervised data is limited and manual labelling expensive. Active learning strategies aim at solving this problem by requiring supervision only on few unlabelled samples, which improve the most model performances after adding them to the training set. Most strategies are based on uncertain sample selection, and even often restricted to samples lying close to the decision boundary. Here we propose a very different approach, taking into consideration domain knowledge. Indeed, in the case of multi-label classification, the relationships among classes offer a way to spot incoherent predictions, i.e., predictions where the model may most likely need supervision. We have developed a framework where first-order-logic knowledge is converted into constraints and their violation is checked as a natural guide for sample selection. We empirically demonstrate that knowledge-driven strategy outperforms standard strategies, particularly on those datasets where domain knowledge is complete. Furthermore, we show how the proposed approach enables discovering data distributions lying far from training data. Finally, the proposed knowledge-driven strategy can be also easily used in object-detection problems where standard uncertainty-based techniques are difficult to apply.

1 INTRODUCTION

Deep Learning (DL) methods have achieved impressive results over the last few years in fields ranging from computer vision to machine translation (LeCun et al., 2015). Most of the research, however, focused on improving model performances, while little attention has been paid to overcome the intrinsic limits of DL algorithms (Marcus, 2018). In particular, in this work we will focus on the amount of data problem. Indeed, deep neural networks need large amounts of labelled data to be properly trained. With the advent of Big Data, sample collection does not represent an issue any more. Nonetheless, the number of *supervised* data in some contexts is limited, and manual labelling can be expensive and time-consuming (Yu et al., 2015). Therefore, a common situation is the unlabelled pool scenario (McCallum & Nigam, 1998), where many data are available, but only some are annotated. Historically, two strategies have been devised to tackle this situation: semi-supervised learning which focus on improving feature representations by processing the unlabelled data with unsupervised techniques (Zhu & Goldberg, 2009); active learning in which the training algorithm indicates which data should be annotated to improve the most its performances. The main assumption behind active learning strategies is that there exists a subset of samples that allows to train a model with a similar accuracy as when fed with all training data. Iteratively, the model indicates the optimal samples to be annotated from the unlabelled pool. This is generally done by ranking the unlabelled samples w.r.t. a given measure and by selecting the samples associated to the highest scores. In this paper, we propose an active learning strategy that compares the predictions over the unsupervised data with the available domain knowledge and exploits the inconsistencies as an index for selecting data to be annotated. Domain knowledge can be generally expressed as First-Order Logic (FOL) clauses and translated into real-valued logic constraints by means of T-Norms (Klement et al., 2013). This formulation has been employed in the semi-supervised learning scenario to improve classifier performance by enforcing the constraints on the unsupervised data (Gnecco et al., 2015; Diligenti et al., 2017). More recently, constraints violation has been effectively used also as a metric to detect adversarial attacks (Melacci et al., 2020). To the best of our knowledge,

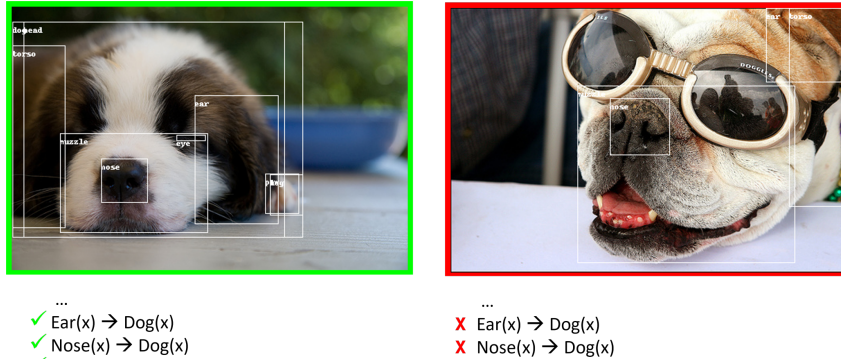


Figure 1: An example of how the proposed method works. Networks predictions on unseen data are compared with a given knowledge (in the form of FOL formulas). Knowledge violation is used as a metric to select samples (figure on the right) which require annotations in an active learning strategy. Images from the *PASCAL-Part* dataset (Chen et al., 2014).

however, domain-knowledge (in the form of logic constraints) violation has never been used as an index in the selection process of an active learning strategy.

A straightforward intuition in active learning is that the algorithm should select the data on which model predictions significantly differ from those produced on the training data. Uncertain sample selection follows this intuition by picking the points on which, e.g., the prediction entropy is high, in contrast to the entropy on the training set which is very low. We also follow this idea by taking into consideration the violation of a given knowledge on unseen data. Indeed, while the logic constraints derived from the domain knowledge are mostly satisfied on the training data, the same does not hold outside the training distribution. To give an example, let us consider the case of Dog image recognition (Figure 1). A model may have learnt on a set of dog images where dogs' muzzle were identifiable. However, on an image coming from an unseen distribution (e.g., a dog belonging to a different species), the model may still recognize the muzzle, but it may not recognize the dog (Figure 1, on the right). The proposed method would detect this image as requiring an annotation because it violates (among others) the logic constraint $\forall x : Muzzle(x) \Rightarrow Dog(x)$. A main assumption of this strategy is that the model does not only output the main object in every image, but it also recognizes some properties of the object or the parts it is composed of. In other words, we define the problem as multi-label classification. Please notice that this assumption does not limit the application to the standard image-classification scenario, since also object-detection can be regarded as a multi-label classification problem (Gong et al., 2019; Zhao et al., 2020). We show that the proposed strategy outperforms the standard uncertain sample selection method, particularly in those contexts where domain-knowledge is rich. We empirically demonstrate that this is mainly due to the fact that the proposed strategy allows discovering data distributions lying far from training data, unlike uncertainty-based approaches. Neural networks, indeed, are known to be over-confident of their prediction, and they are generally unable to recognize samples lying far from the training data distribution. This issue, beyond exposing them to adversarial attacks (Szegedy et al., 2013; Goodfellow et al., 2014), prevents uncertainty-based strategies from detecting these samples as points that would require an annotation. On the contrary, even though a neural network may be confident of its predictions, the interaction between the predicted classes may still offer a way to spot out-of-the-distribution samples. As reported in the example above, a neural network may predict with high level of confidence the presence of the muzzle in the image and, as certainly, it may not recognize the dog. The missing interaction of the two classes, however, allows spotting an incoherent prediction. Finally, as anticipated above, the Knowledge-driven Active Learning (KAL) strategy can be also employed in the object-detection context where standard uncertainty-based ones are difficult to apply (Choi et al., 2021; Haussmann et al., 2020).

In Section 2 the proposed method is explained in details, with first an example on inferring the XOR operation and then contextualized in a more realistic active learning domain; the experimental results on three datasets are described in Section 3, comparing the proposed technique with a standard active learning strategy; in Section 4 the related work is briefly resumed; finally, in Section 5 some final comments on the work are reported with possible extensions and future works.

2 KNOWLEDGE-DRIVEN ACTIVE LEARNING

In this paper, we focus on multi-label classification problems with c classes, in which each input $x \in X$ is associated to one or more classes and d represent the input dimensionality. We also consider the case in which additional *domain knowledge* is available for the problem at hand, which may be represented by a set of relationships existing either between the input features x and the classes or simply among the classes. How to exploit such knowledge to select the best points for training a classifier is the main subject of this section, and it follows the principles presented in (Gori & Melacci, 2013; Gnecco et al., 2015; Diligenti et al., 2017). Using domain knowledge in the selection process provides precious information to define a criterion for identifying examples on which the model requires supervision.

Let us consider the classification problem $f : X \rightarrow Y$, where $X \subseteq \mathbb{R}^d$ represents the feature space which may also comprehend non-structured data (e.g. input images) and $Y \subseteq \{0, 1\}^c$ is the output space composed of $c \geq 1$ dimensions. More precisely, we consider a vector function $f = [f^1, \dots, f^c]$, where each function f^i predicts the probability that x belongs to the i -th class. When considering an object-detection problem, for a given class and a given image, we consider as class membership probability the maximum score value among all predicted bounding boxes around objects belonging to that class. Formally, $f^i(x_j) = \max_{s_h \in \mathcal{S}^i(x_j)} s_h(x_j)$ where $\mathcal{S}^i(x_j)$ is the set of the confidence scores of the bounding boxes predicting the i -th class for sample x_j . Obviously, in case a certain class is not predicted in any of the bounding box, we set $f^i(x_j) = 0$. By considering each f^i as a logic predicate, First-Order Logic (FOL) becomes the natural way of describing relationships between data and classes or only among the classes. Let us illustrate this with an example of the first case, $\forall x \in X, x^1(x) \wedge x^2(x) \Rightarrow f(x)$, where $x^1(x), x^2(x)$ respectively represent the logic predicates associated to the first and second feature, and meaning that when both predicates are true also the output function $f(x)$ needs to be true. Now, let us provide an example for the second case, $\forall x \in X, f^v(x) \wedge f^z(x) \Rightarrow f^u(x)$, for some v, z, u , meaning that the intersection between the v -th class and the z -th class is always included in the u -th one. Let us also consider the case in which there exists $X_s \subset X$, representing the portion of input data already associated to an annotation $y_i \in Y_s \subset Y$. We define with n the dimensionality of the starting set of labelled data. At each iteration, a set of p samples $[x_1, x_2, \dots, x_p] = X_p \subset X_u \subset X$ is selected by the active learning strategy to be annotated, being X_u the set of (still) unlabelled input data. This process is repeated for a number of iterations b , after which the training terminates.

2.1 CONVERTING DOMAIN-KNOWLEDGE INTO LOSS FUNCTIONS

The Learning from Constraints framework (Gnecco et al., 2015; Gori & Melacci, 2013; Diligenti et al., 2017) define a way to convert domain knowledge into logic constraints and how to use them on the learning problem. Among a variety of other type of constraints (see, e.g., Table 2 in (Gnecco et al., 2015)), it studies the process of handling FOL formulas so that they can be both injected into the learning problem (in Semi-Supervised learning) or used as a knowledge verification measure (as in (Melacci et al., 2020) and in the proposed method). Based on this assumption, an active learning strategy can detect whether the predictions made by the model on out-of-sample data are coherent with the domain knowledge or not. Going into more detail, the FOL formulas representing the domain knowledge are converted into numerical constraints using the Triangular Norms (T-Norms, (Klement et al., 2013)). These binary functions generalize the conjunction operator \wedge and offer a way to mathematically compute the satisfaction level of a given rule. Following the previous example, $x^1(x) \wedge x^2(x) \Rightarrow f(x)$ is converted into a bilateral constraint $\phi(f(x)) = 1$ that, by employing the product T-Norm, is $1 - x^1(x)x^2(x)(1 - f(x)) = 1$. With $\hat{\phi}(f(x)) = 1 - \phi(f(x))$ we indicate the loss function associated to the bilateral constraints, which measures the level of satisfaction of the given constraints and has its minimum value in zero. Again, recalling the previous example, the associated loss function would be $x^1(x)x^2(x)(1 - f(x))$, which indeed is satisfied when either $x^1(x)$ or $x^2(x)$ are zeros or $f(x)$ is approximately one. Finally, the loss function considering all the available FOL formulas \mathcal{K} for the given problem is computed to select the points which violates most the constraints. In particular, this is done by aggregating the losses of all the corresponding constraints for all the samples $x \in X_u$:

$$\text{KAL} : \quad [x_1, x_2, \dots, x_p] = \arg \max_{x \subset X_u} \sum_k^K \hat{\phi}_k(f(x)) \quad (1)$$

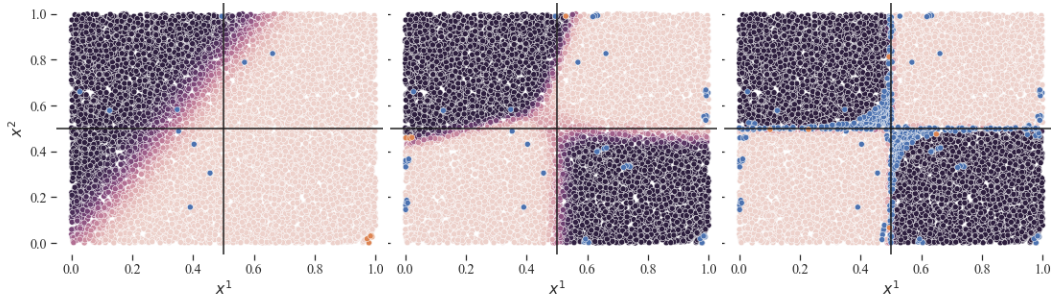


Figure 2: A visual example on the *XOR-like* problem, showing the principles of the KAL strategy. We depict network predictions with different colour degrees (light colours negative predictions, dark colours positive prediction). In blue, we depict the points selected in previous iterations, in orange those selected at the current iteration. Black lines at $x^1 = 0.5$ and $x^2 = 0.5$ are reported only for visualization purposes. From left to right, the situation at the 1st, 10th and 100th iteration.

2.2 A PERFECT EXAMPLE: THE *XOR-like* PROBLEM

A well-known problem in machine learning is the inference of the eXclusive OR (XOR) operation. To show the working principles of the proposed approach, we propose here a variant of this experiment in which a neural network has to learn a XOR-like operation from a distribution of non-boolean samples. More in details, we sampled 10000 points from the distribution $x \in [0, 1]^2$, and we assigned a label $y(x)$ as following:

$$y(x) = \begin{cases} 1, & \text{for } 0.5 \leq x^1 < 1, 0 < x^2 \leq 0.5, \\ 1, & \text{for } 0 < x^1 \leq 0.5, 0.5 \leq x^2 < 1, \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

Also, we can express the XOR operation through a FOL formula $(x^1 \wedge \neg x^2) \vee (\neg x^1 \wedge x^2) \Leftrightarrow f^1$. For the sake of clarity, here we drop the argument (x) of the logic predicates within logic formulas. As seen before, through the T-Norm operation we can convert the logic rule into a numerical constraint, and we can calculate its violation through the loss functions:

$$\begin{aligned} \hat{\phi}_1 &= f(1 - x^1(1 - x^2))(1 - x^2(1 - x^1)), \\ \hat{\phi}_2 &= (1 - f)(1 - (1 - x^1(1 - x^2))(1 - x^2(1 - x^1))), \end{aligned} \quad (3)$$

each one representing one direction of the double implication. For an automatic derivation of the loss associated to the violation of a certain rule, please refer to Marra et al. (2019). The Knowledge-driven Active Learning strategy can therefore compute for each sample the associated loss (Eq. 3) and at each iteration select the point that mostly violate them (Eq. 1).

In Fig. 2, we reported an example of the proposed strategy starting from $n = 10$ labelled data and by selecting $p = 5$ points at each iteration for a total of $k = 200$ iterations. We depicted the network predictions with different colour degrees ranging from the light beige (negative predictions) to black (positive predictions), the points selected in previous iterations in blue and the points selected at the current iteration in orange. It is easy to notice that already after 10 iterations (figure at the centre) the network has mostly learnt the correct data distribution. After 100 iterations (figure on the right), the proposed strategy has correctly selected most of the points along the decision boundaries, allowing the network to already solve the problem (accuracy $\sim 99\%$, see Sec. 3). At last, notice how the random starting points (figure on the left) did not cover all the data distributions (no blue points in the right-bottom quadrant). Nonetheless, KAL correctly selects points from the unseen distribution (orange points), since they are violating at most the given knowledge.

¹Practically, the predicate x^i is obtained by applying a sigmoid function over the i -th feature of the input sample x , $\sigma(x) = \frac{1}{1 + e^{-k(x - 0.5)}}$, where k is a temperature parameter determining the steepness of the sigmoid (set to $k = 10$), and -0.5 is added to the argument x to centre the function at 0.5.

2.3 REAL-LIFE SCENARIO: PARTIAL KNOWLEDGE AND DIFFERENT TYPE OF RULES

It is clear that, in the case of the *XOR-like* problem, the knowledge is complete, in the sense that this rule already explains the learning problem. However, the purpose of that simple experiment is only to show the potentiality of the proposed approach in integrating the available symbolic knowledge into a learning problem. In a real-life scenarios, such a situation is unrealistic, but still we might have access to some useful knowledge that may help us to solve more quickly a given learning problem. More precisely, we consider multi-label classification problems, where classes stand for concepts as much as attributes, with annotations involving several semantic levels. This is an increasingly common task in fields ranging from medical domain (Chen et al., 2019), to cultural heritage (Bobasheva et al., 2021), and fine-grained classification (Wah et al., 2011), to cite a few.

In this type of scenario and when considering a given domain-knowledge, there exists a variety of FOL-formulas to express the relations between the classes. Let us consider the classes as entities in an Entity-Relationship (ER) model (Chen, 1976). In multi-label classification, the domain knowledge generally consists of many-to-many relationships with two set of classes, where the entities from both sets can be involved in many relationships. To translate this into a set of FOL rules, we generally make use of an implication rule between instances of the two entities. Let us consider, as an example, a Man-vs-Dog classification: we might know that one of the main object (e.g., a dog, belonging to the first set of entity) is composed of several parts (e.g., a muzzle, a body, a tail, four paws, from the second set of entities). A straightforward translation of this compositional property into a FOL rule might be $\forall x, Dog(x) \Rightarrow Muzzle(x) \vee Body(x) \vee Tail(x) \vee Paws(x)$. However, formulating the composition in the opposite way is also formally correct, i.e., from the parts to the main object (e.g., $\forall x, Muzzle(x) \Rightarrow Dog(x)$). These two types of knowledge in the following will be respectively referred as **type a**) and **type b**) rules. When considering the previous classification problem, however, it is also true that at least a human or a dog need to be present in each image, formally $\forall x, Dog(x) \vee Man(x)$. Finally, if the dataset is properly labelled, at least one of the object-parts needs to be present, therefore $\forall x, Head(x) \vee Arm(x) \vee Body(x) \dots Muzzle(x) \vee Paws(x) \vee Tail(x)$. We refer to the latter as **type c**) and **type d**) formulas. While some of the previously formulas may be more useful than others, we empirically discovered that we achieve the best results when all the types of formulas are given (see *PASCAL-Part* experiment in Sec 3.4).

2.4 ADDING DIVERSITY SAMPLING

As it has been anticipated in the introduction, uncertainty-based method in DL may not be very effective in case they are not paired with a diversity sampling strategy, similarly to Brinker (2003) introducing diversity in Tong & Koller (2001) margin-based approach. Also in the case of KAL this holds true, given a set of rules \mathcal{K} , the proposed method might select p samples all violating the same rule $\phi_k(f(x))$. Even though a neural network may need different samples to learn a novel distribution of data, picking a batch of samples belonging to the same distribution might be a poor strategy and slow down the overall training process. In order to avoid this issue, we decided to select a maximum number of samples r violating a certain rule k . More precisely, we group samples $x \in X_u$ according to the rule they violate the most, and we select a maximum number r of samples from each cluster (still following the raking given by Eq. 1).

3 EXPERIMENTS

In this work, we considered three different scenarios, comparing the proposed technique with other active learning strategies. More precisely, we evaluated the various methods on the inference of the *XOR-like* problem (already introduced in Section 2.2), on an image-classification scenario (on the *Animals* dataset), and on an object-recognition task (on the *PASCAL-Part* dataset (Chen et al., 2014)). In Section 3.1 we report the details regarding each experimental problems; in Section 3.2 the types of knowledge employed in the experiments is reported; in Section 3.3; the compared methods are briefly described and analysed; finally, in Section 3.4 qualitative and quantitative analysis of the different active learning strategies are reported for the three learning problems. All the experiments have been repeated three times, starting from a different weight initialization of the models. The code required to run the experiments is published on a publicly available repository and it is provided in the supplementary material. Also, a simple code example is reported in Appendix A showing how to train a model on the *XOR-like* problem following the KAL strategy.

3.1 EXPERIMENTAL DETAILS

The problem of inferring the *XOR-like* operation has been already introduced in Section 3.4: it is an artificial dataset consisting of 10000 samples $x \in X^2$, mapped the corresponding label $y \in Y^1$ as in Eq. 2. A neural network $f : X^2 \rightarrow Y^1$ is used to solve the task. It is equipped with a single hidden layer of 100 neurons and Rectified Linear Unit (ReLU) activation, and a single output neuron with sigmoid activation. It has been trained with an AdamW optimizer (Loshchilov & Hutter, 2017) for 100 epochs at each iteration, with a learning rate $\eta = 10^{-2}$. Standard cross-entropy loss has been used to enforce f to learn the available supervisions. By starting from $n = 10$ samples, we added $p = 5$ labelled samples at each iteration for a total of $b = 198$ iterations, resulting in 1000 supervisions at the last iteration. The accuracy reported in Figure 4a corresponds to the model accuracy over the whole pool of data, i.e. over both the labelled and the unlabelled samples.

For more real-life style problems, we have considered two datasets where the domain-knowledge is partial and more complex than the above XOR experiment.

The *Animals*² dataset is a collection of 8287 images of animals, taken from the ImageNet database². The task consists in the classification of 7 main classes (Albatross, Giraffe, Cheetah, Ostrich, Penguin, Tiger, Zebra) and 26 representing animal attributes (e.g. Mammal, Fly or Lay_Eggs), for a total of $c = 33$ classes. Images have been resized to a dimension $d = 256 \times 256$ pixels. A Resnet50 Convolutional Neural Network (He et al., 2015) has been employed to solve the task $f : X^d \rightarrow Y^c$. Going into more details, a transfer learning strategy has been employed: the network f was pretrained on the ImageNet dataset and only the last fully connected layer has been trained (from scratch) on the *Animals* dataset. Again, an AdamW optimizer is considered with a learning rate $\eta = 10^{-3}$ for 1000 epochs at each iteration with standard cross-entropy loss. Owing to the increased difficulty of the problem, we started with $n = 100$ labelled samples, and we added $p = 50$ samples each time for $b = 98$ iterations, for a total of 5000 labelled samples. For diversity sampling, at each iteration we pick a maximum of $r = 20$ samples violating the same rule. Since *Animals* is a multi-label classification problem, in Figure 4b we reported the F1 score of the model when varying the number of supervised training samples.

The *PASCAL-Part* dataset³ is composed of 10103 images of varying size, depicting objects (MAN, DOG, etc.) and object-parts (Head, Muzzle, Tail, etc.). We preprocessed the dataset following the approach of Serafini & Garcez (2016), merging specific parts into unique labels. Furthermore, we have divided original part classes describing very different objects into different classes (e.g., body to bottle-body and aeroplane-body), leading to $c = 66$ classes, out of which 16 are main objects. In this dataset, labels are given in the form of segmentation masks. A Faster R-CNN network (Ren et al., 2016) is trained on the bounding boxes extracted from each mask, the leftmost and highest pixels are used for the first coordinate and the opposites for the second one. Owing to the computational complexity of the task, the model has been trained for 50 epochs at each iteration, with an SGD optimizer with momentum and a learning rate schedule with an initial value $\eta = 3 \cdot 10^{-3}$ decreasing by 0.3 every 20 epochs. In this case, we started with $n = 500$ labelled examples and by adding $p = 50$ samples for $b = 10$ iterations for a total of 1000 supervisions. At each iteration we pick a maximum of $r = 5$ samples violating the same rule. In Figure 4c we reported the growth of the mean Average Precision (mAP) of the model averaged 10 times with Intersection over Union (IoU) ranging from 0.5 to 0.95.

3.2 EMPLOYED KNOWLEDGE

As anticipated, in the *XOR* problem, the rule employed for the KAL strategy is $(x^1 \wedge \neg x^2) \vee (\neg x^1 \wedge x^2) \Leftrightarrow f$. In the case of *Animals*, instead, the employed knowledge is a simple collection of 16 FOL formulas, defined by Winston & Horn (1986) as a benchmark. They involve relationships between animals and their attributes, such as $\forall x \text{ Fly}(x) \wedge \text{Lay_Eggs}(x) \Rightarrow \text{Bird}(x)$ (mostly *type b*) rules, following the notation introduced in Section 2.3). To this collection of rules, we have also added a *type c*) and a *type d*) rule, i.e. a mutual exclusive disjunction among the animal classes (only one animal is present in each image) and a standard disjunction over the animal attributes (each animal has more than one attribute). On the contrary, on *PASCAL-Part* FOL rules we have devised a set of rules covering all the types presented in Section 2.3, therefore we have *type a*) rules, listing the parts

²*Animals*: <http://www.image-net.org/>

³*PASCAL-Part*: <http://roozbehm.info/pascal-parts/pascal-parts.html>.

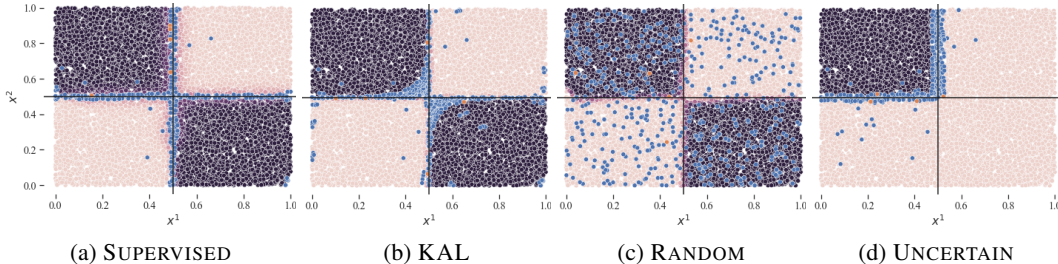


Figure 3: A comparison of the sample selection process on the *XOR* task (starting from the same points as in Figure 2). The proposed method mostly selects data along the decision boundaries, similarly to the supervised one. Notice how the uncertainty-based strategy is not capable of discovering novel data distribution (right-bottom quadrant).

belonging to a certain object, (e.g. $\text{Motorbike}(x) \Rightarrow \text{Wheel}(x) \vee \text{Headlight}(x) \vee \text{Handlebar}(x) \vee \text{Saddle}(x)$), *type b*) listing all the objects in which a part can be found (e.g., $\text{Handlebar}(x) \Rightarrow \text{Bicycle}(x) \vee \text{Motorbike}(x)$), *type c*) rule, involving a disjunction of all the main classes and *type d*), involving a disjunction of all the object-parts, for a total of 62 rules employed. A complete list of the rules employed in both experiments is reported in Appendix C.

3.3 COMPARED METHODS

We compared KAL to three other active learning strategies: standard UNCERTAIN-sample selection, RANDOM selection, but also a SUPERVISED strategy, i.e. a utopian active learning strategy which consists in evaluating the supervision loss on the whole pool of the unlabelled samples. Obviously, this latter is an unfeasible active learning strategy because it would require to already have the labels of all the samples in the unlabelled pool. However, as the RANDOM selection can be regarded as a lower-bound, we can consider SUPERVISED as a possible upper-bound of the quality of an active learning strategy. More precisely, it is computed as the cross entropy between the predictions of the network f and the actual labels y :

$$\text{SUPERVISED} : [x_1, x_2, \dots, x_p] = \arg \max_{x \in X_u, y \in Y_u} \sum_{i=1}^c -(y^i \cdot \log(f^i(x)) + (1 - y^i) \cdot \log(1 - f^i(x))), \quad (4)$$

where Y_u is the set of labels associated to the samples X_u . Regarding the UNCERTAIN-sample selection, among other possibilities, we consider here the closest samples to the decision boundaries. Formally:

$$\text{UNCERTAIN} : [x_1, x_2, \dots, x_p] = \arg \max_{x \in X_u} \|f(x) - 0.5\|, \quad (5)$$

where an L_1 norm is employed to compute the distance from the decision boundary (0.5).

3.4 EXPERIMENTAL RESULTS

For a qualitative evaluation of the proposed approach, in Figure 3 we reported the samples selected by the various strategies at the 100th iteration on the *XOR* task, starting from the same randomly selected samples of Figure 2. Further figures showing the training prediction at different iteration are reported in Appendix B for each of the compared methods. In this context, where the knowledge of the domain is complete, it is obvious how the proposed approach mostly selects the samples along the actual decision boundaries (i.e., $x_1 = 0.5$ and $x_2 = 0.5$) and is capable of discovering data distributions that are not represented by the sting samples. This behaviour mimics the one obtained when selecting the points through the SUPERVISED technique. On the contrary, when employing UNCERTAIN sample selection, novel data distributions are difficult to discover, leading to poorer performances. Indeed, as shown in Figure 4a, UNCERTAIN sample selection requires many more points on average to fully cover all data distributions. It is also worth noticing how with RANDOM selection the model does not reach 100 % accuracy even after 1000 samples, highlighting hence the need of an active learning strategy to perfectly solve this task. A similar situation is also repeated on the *Animals*' classification task, as reported in Figure 4b which analyses the growth of the F1 score

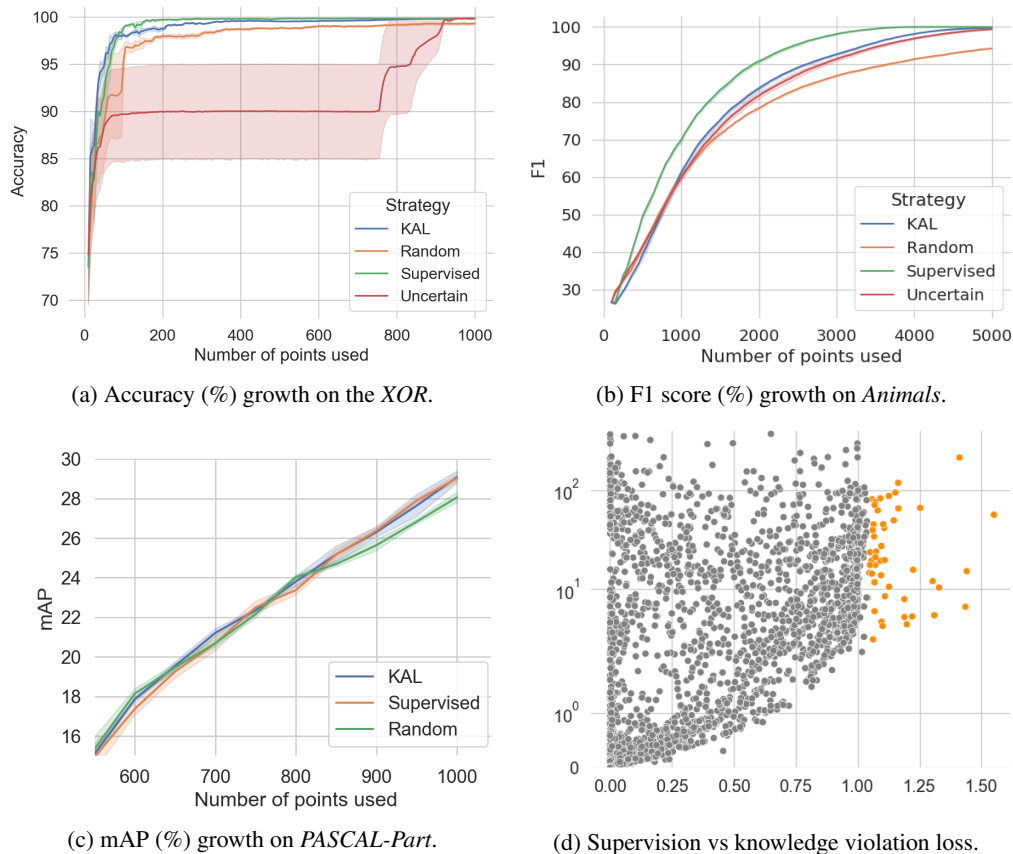


Figure 4: a), b), c) Performance growth on the three experiments when increasing the number of labelled samples. d) Scatter plot of the model supervision loss vs the knowledge violation loss on the *Animals*'s problem at the 50th iteration. In orange, the point selected by the KAL strategy.

of the model. Even in this case, where the given knowledge is very simple, the proposed method outperforms UNCERTAIN sample selection. The distance from the SUPERVISED approach, however, is larger than in the previous case. At last, in Figure 4c, we report the mAP of the model in the very difficult *PASCAL-Part* task,⁴ where many objects and object-parts need to be recognized within the same image. We can notice how, particularly when increasing the number of labelled points (850-1000), the proposed approach improve the performance of the model with respect to a RANDOM selection. Furthermore, the more structured knowledge available in this case allows the proposed approach to accurately select the best samples, achieving similar performances to those obtained with a SUPERVISED technique. As anticipated, it is not straightforward applying UNCERTAIN (Eq. 5) sample selection to the object-detection context, thus, in this case, we restricted the comparison to the other methods only.

We further investigated these results analysing the correlation between the supervision loss (argument of Eq. 4) and the knowledge-violation loss (argument of Eq. 1), on the *Animals*' problem. In Figure 4d, we report a scatter plot of these two metrics calculated over the unlabelled samples at the 50th training iteration. Samples selected by the KAL strategy are depicted in orange, in grey the remaining samples in the unlabelled pool. This scatter plot shows a high-level of correlation of the two distributions: on average, predictions associated with high supervision loss are also associated with high knowledge violation. In particular, we can notice that the points selected by the proposed method are on average wrong model predictions (supervision loss greater than 10). In other words, this figure suggests that knowledge violation can indeed be used as an index to select the points on which the model most likely requires supervision, validating the overall approach.

⁴A Faster-RCNN trained following a standard supervised learning technique on 90% of the dataset and evaluated on the remaining 10% has a test mAP $\sim 41.5\%$ after 200 epochs.

4 RELATED WORK

It has been pondered that human beings' cognition mainly consists in two different tasks: perceiving the world and reasoning over it (Solso et al., 2005). While in humans they take place at the same times, in artificial intelligence these two tasks are separately conducted by machine learning and logic programming. It has been argued that joining these two fields (to create a so-called hybrid model) may overcome some of the most important limits of deep learning, among which the data hungry issue (Marcus, 2018). In the literature, there exists a variety of proposals aiming at this objective, ranging from Statistical Relational Learning (SRL) (Koller et al., 2007) and Probabilistic Logic Programming (De Raedt & Kimmig, 2015) which focus on integrating learning with logic reasoning, to enhanced networks focusing on relations (Santoro et al., 2017) or with external memories (Graves et al., 2016). To the best of our knowledge, however, none of these methods can be directly applied in the standard active learning scenario. On the contrary, we have shown that the learning from constraints (Gnecco et al., 2015; Diligenti et al., 2017) framework can be naturally leveraged to devise active learning strategies in context where a domain-knowledge is available.

Devising an active learning strategy is not an easy task. As previously introduced, an approach could be to simply select the points on which the model is just wrong about. Since this is obviously not possible, in the literature, two main approaches have been followed: uncertainty sampling which select the data on which the model is the least confident, possibly extended with diversity sampling which maximizes the data distribution exploration among selected samples; curriculum learning which instead focuses first on easy samples then extending the training set to incorporate more and more difficult ones while also targeting more diversity. Standard uncertain strategies consist in choosing samples that maximize the prediction entropy (Houlsby et al., 2011; Cao & Tsang, 2021), the distance from the hyperplane in SVM (Schohn & Cohn, 2000), or the variation ratio in Query-by-committee with ensemble methods (Burbidge et al., 2007; Ducoffe & Precioso, 2017; Beluch et al., 2018). Establishing prediction uncertainty is more difficult with DL models. Indeed, they generally tend to be over-confident, particularly when employing softmax activation functions (Thulasidasan et al., 2019). Furthermore, there is no easy access to the distance to the decision boundary as for SVM, so it needs to be computed. This problem has been tackled by devising different uncertain strategies, such as employing Bayesian Neural Network with Monte Carlo Dropout (Gal et al., 2017), predicting the loss associated to each sample (Yoo & Kweon, 2019), or calculating the minimum distance required to create an adversarial example (Ducoffe & Precioso, 2018). As pointed out by Pop & Fulop (2018), however, uncertain strategy alone may choose the same categories many times and may create unbalanced datasets. In order to solve this, uncertain sample selection needs to be coupled with diversity sampling strategies. Diversity is generally obtained by preferring batches of data maximizing the mutual information between model parameters and predictions (Kirsch et al., 2019), or core-set points (Sener & Savarese, 2018), or, also, samples nearest to k-means cluster centroids (Zhdanov, 2019). At last, by considering gradient parameters with respect to the predicted category, one can compute at the same time prediction uncertainty (by selecting samples with higher gradient norm) and sample diversity (by maximizing the diversity among the selected samples gradients) (Ash et al., 2019).

5 CONCLUSIONS

In this paper we proposed an active learning strategy (KAL) driven by knowledge consistency principles. The KAL strategy inspects model predictions on unseen data to detect those violating the logic constraints. The latter are extracted from an available domain-knowledge by means of T-Norm. The performance of a model equipped with such a strategy outperforms standard uncertainty-based approaches. As future work, KAL paves the way for novel online learning methods in which the model keep learning only on those points where the knowledge is violated. The proposed approach could be refined by asking for supervision only for the predicates involved in the violated rules, reducing further the number of required labelled data and leading to a more balanced training set improving the prediction on smaller classes. At last, in case no knowledge is available on a certain problem, a first idea could be to pair the KAL strategy with the method proposed in Ciravegna et al. (2021), where FOL explanations of network predictions are extracted on training data, to continuously check whether the knowledge learnt on the training distribution is also valid on unseen data.

ACKNOWLEDGMENTS

This work has been partially supported by TAILOR, a project funded by EU Horizon 2020 research and innovation programme under GA No 952215.

REFERENCES

- Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*, 2019.
- William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9368–9377, 2018.
- Anna Bobasheva, Fabien Gandon, and Frédéric Precioso. Learning and Reasoning for Cultural Metadata Quality. *Journal on Computing and Cultural Heritage*, 2021. URL <https://hal.archives-ouvertes.fr/hal-03363442>.
- Klaus Brinker. Incorporating diversity in active learning with support vector machines. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pp. 59–66, 2003.
- Robert Burbidge, Jem J Rowland, and Ross D King. Active learning for regression based on query by committee. In *International conference on intelligent data engineering and automated learning*, pp. 209–218. Springer, 2007.
- Xiaofeng Cao and Ivor W. Tsang. Bayesian active learning by disagreements: A geometric perspective, 2021.
- Peter Pin-Shan Chen. The entity-relationship model—toward a unified view of data. *ACM transactions on database systems (TODS)*, 1(1):9–36, 1976.
- Pingjun Chen, Linlin Gao, Xiaoshuang Shi, Kyle Allen, and Lin Yang. Fully automatic knee osteoarthritis severity grading using deep neural networks with a novel ordinal loss. *Computerized Medical Imaging and Graphics*, 75:84–92, 2019.
- Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts, 2014.
- Jiwoong Choi, Ismail Elezi, Hyuk-Jae Lee, Clement Farabet, and Jose M. Alvarez. Active learning for deep object detection via probabilistic modeling, 2021.
- Gabriele Ciravegna, Pietro Barbiero, Francesco Giannini, Marco Gori, Pietro Lió, Marco Maggini, and Stefano Melacci. Logic explained networks. *arXiv preprint arXiv:2108.05149*, 2021.
- Luc De Raedt and Angelika Kimmig. Probabilistic (logic) programming concepts. *Machine Learning*, 100(1):5–47, 2015.
- Michelangelo Diligenti, Marco Gori, and Claudio Sacca. Semantic-based regularization for learning and inference. *Artificial Intelligence*, 244:143–165, 2017.
- Melanie Ducoffe and Frédéric Precioso. Active learning strategy for cnn combining batchwise dropout and query-by-committee. In *ESANN*, 2017.
- Melanie Ducoffe and Frederic Precioso. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*, 2018.
- Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data, 2017.
- Giorgio Gnecco, Marco Gori, Stefano Melacci, and Marcello Sanguineti. Foundations of support constraint machines. *Neural computation*, 27(2):388–480, 2015.

-
- Tao Gong, Bin Liu, Qi Chu, and Nenghai Yu. Using multi-label classification to improve object detection. *Neurocomputing*, 370:174–185, 2019. ISSN 0925-2312. doi: <https://doi.org/10.1016/j.neucom.2019.08.089>. URL <https://www.sciencedirect.com/science/article/pii/S0925231219312585>.
- Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- Marco Gori and Stefano Melacci. Constraint verification with kernel machines. *IEEE Trans. Neural Networks Learn. Syst.*, 24(5):825–831, 2013. doi: 10.1109/TNNLS.2013.2241787. URL <https://doi.org/10.1109/TNNLS.2013.2241787>.
- Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwinska, Sergio Gomez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John P. Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538:471–476, 2016.
- Elmar Haussmann, Michele Fenzi, Kashyap Chitta, Jan Ivanecy, Hanson Xu, Donna Roy, Akshita Mittel, Nicolas Koumchatzky, Clement Farabet, and Jose M. Alvarez. Scalable active learning for object detection, 2020.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.
- Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning, 2011.
- Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning, 2019.
- Erich Peter Klement, Radko Mesiar, and Endre Pap. *Triangular norms*, volume 8. Springer Science & Business Media, 2013.
- Daphne Koller, Nir Friedman, Sašo Džeroski, Charles Sutton, Andrew McCallum, Avi Pfeffer, Pieter Abbeel, Ming-Fai Wong, Chris Meek, Jennifer Neville, et al. *Introduction to statistical relational learning*. MIT press, 2007.
- Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.
- Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- Gary Marcus. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*, 2018.
- Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, and Marco Gori. Lyrics: A general interface layer to integrate logic inference and deep learning. *arXiv preprint arXiv:1903.07534*, 2019.
- Andrew Kachites McCallumzy and Kamal Nigamy. Employing em and pool-based active learning for text classification. In *Proc. International Conference on Machine Learning (ICML)*, pp. 359–367. Citeseer, 1998.
- Stefano Melacci, Gabriele Ciravegna, Angelo Sotgiu, Ambra Demontis, Battista Biggio, Marco Gori, and Fabio Roli. Domain knowledge alleviates adversarial attacks in multi-label classifiers. *arXiv preprint arXiv:2006.03833*, 2020.
- Remus Pop and Patric Fulop. Deep ensemble bayesian active learning : Addressing the mode collapse issue in monte carlo dropout via ensembles, 2018.
- Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2016.

-
- Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning, 2017.
- Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. *Machine Learning-International Workshop then Conference*, 10 2000.
- Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach, 2018.
- Luciano Serafini and Artur d’Avila Garcez. Logic tensor networks: Deep learning and logical reasoning from data and knowledge. *arXiv preprint arXiv:1606.04422*, 2016.
- Robert L Solso, M Kimberly MacLin, and Otto H MacLin. *Cognitive psychology*. Pearson Education New Zealand, 2005.
- Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.
- Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66, 2001.
- C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.
- Patrick Henry Winston and Berthold K Horn. Lisp. 1986.
- Donggeun Yoo and In So Kweon. Learning loss for active learning, 2019.
- Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.
- Zhen Zhao, Yuhong Guo, Haifeng Shen, and Jieping Ye. Adaptive object detection with dual multi-label prediction. In *European Conference on Computer Vision*, pp. 54–69. Springer, 2020.
- Fedor Zhdanov. Diverse mini-batch active learning, 2019.
- Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.

A SOFTWARE

The Python code and the scripts used for the experiments, including parameter values and documentation, is freely available under Apache 2.0 Public Licence from a GitHub repository. The proposed approach requires only a few lines of code to train a model following the KAL strategy, as we sketch in the following code example (Listing 1).

```
1 tot_points = 10000
2 first_points = 10
3 n_points = 5
4 n_iterations = 198
5
6 # Generating data for the xor problem
7 x = np.random.uniform(size=(tot_points, 2))
8 y = np.ndarray.astype(((x[:, 0] > 0.5) & (x[:, 1] < 0.5)) |
9                       ((x[:, 1] > 0.5) & (x[:, 0] < 0.5))), float)
10 x_t = torch.as_tensor(x, dtype=torch.float)
11 y_t = torch.as_tensor(y, dtype=torch.float)
12
13 # Defining constraints as product t-norm of the FOL rule expressing the XOR (x1 & ~x2
14   ) | (x2 & ~x1) <=> f
15 def calculate_constraint_loss(x_continue: torch.Tensor, f: torch.Tensor):
16     discrete_x = steep_sigmoid(x_continue).float()
17     x1 = discrete_x[:, 0]
18     x2 = discrete_x[:, 1]
19     cons_loss1 = f * ((1 - (x1 * (1 - x2))) * (1 - (x2 * (1 - x1))))
20     cons_loss2 = (1 - f) * (1 - ((1 - (x1 * (1 - x2))) * (1 - (x2 * (1 - x1)))))
21     return cons_loss1 + cons_loss2
22
23 # Constrained Active learning strategy
24 # We take the p samples that most violate the constraints and are among available idx
25 def cal_selection(not_avail_idx: list, c_loss: torch.Tensor, p: int):
26     c_loss[torch.as_tensor(not_avail_idx)] = -1
27     cal_idx = torch.argsort(c_loss, descending=True).tolist()[ :p]
28     return cal_idx
29
30 # Few epochs with n randomly selected data
31 net = MLP(2, 100)
32 first_idx = np.random.randint(0, x.shape[0] - 1, first_points).tolist()
33 train_loop(net, x_t, y_t, first_idx)
34
35 preds_t = net(x_t)
36
37 cons_loss = calculate_constraint_loss(x_t, preds_t)
38 available_idx = [*range(tot_points)]
39 used_idx = first_idx
40
41 # Active Learning with KAL strategy for n_iterations
42 for n in range(1, n_iterations + 1):
43     available_idx = list(set(available_idx) - set(used_idx))
44     active_idx = cal_selection(used_idx, cons_loss, n_points)
45     used_idx += active_idx
46
47     # train for 50 epochs the MLP on the used idx
48     train_loop(net, x_t, y_t, used_idx, epochs=50)
49
50 preds_t = net(x_t).squeeze()
51 accuracy = (preds_t > 0.5).eq(y_t).sum().item() / y_t.shape[0] * 100
52 cons_loss = calculate_constraint_loss(x_t, preds_t)
```

Listing 1: KAL code - Example on the XOR problem.

B TRAINING EVOLUTIONS ON THE *XOR-like* PROBLEM

In this appendix, we report further snapshots of the training process in Figure 5. They depict for each of the compared method the model predictions at different iterations, similarly to what it has been shown in Figure 2 for the KAL strategy (which are also reported for comparison). As it has already been noticed, UNCERTAIN is the only method unable to discover the data distribution in the right-bottom angle (for which no samples have been drawn during the initial random sampling) even after 100 iterations. Also, it is interesting to notice how the sampling selection performed by the proposed approach resemble that made by the SUPERVISED one. For the complete animations showing the evolution of the training at each iteration for each method, please check the public GitHub repository and the supplementary materials.

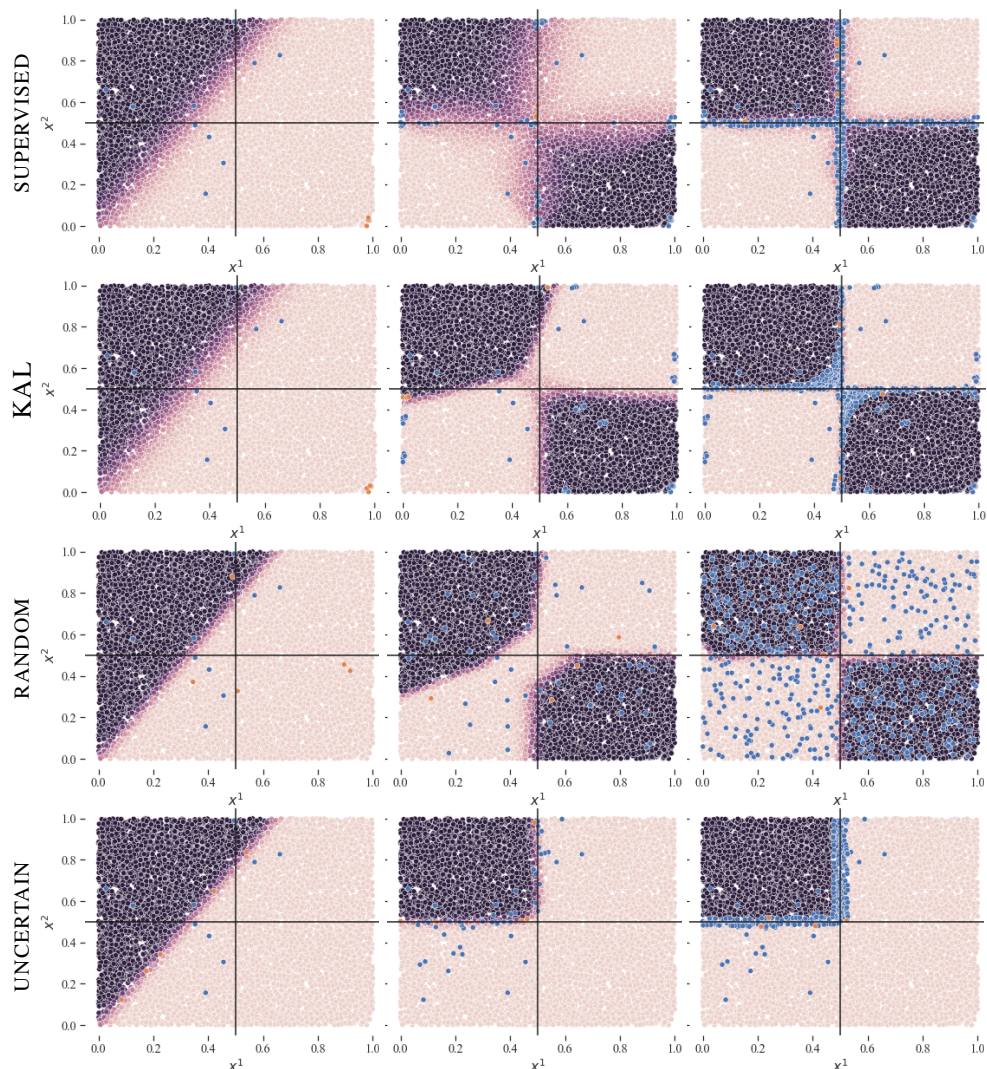


Figure 5: A visual example on the *XOR-like* problem, showing how the training evolves in each of the compared strategy. We depict network predictions with different colour degrees (light colours negative predictions, dark colours positive prediction). In blue, we depict the points selected in previous iterations, in orange those selected at the current iteration. Black lines at $x^1 = 0.5$ and $x^2 = 0.5$ are reported only for visualization purposes. From left to right, the situation at the 1st, 10th and 100th iteration.

C LIST OF EMPLOYED RULES

In each of the presented task, we need to predict a set of classes that can be associated to logic predicates. The available domain knowledge is converted into FOL formulas involving such predicates. The formulas representing the domain knowledge on the ANIMALS and PASCAL-Part dataset are reported in Table 1, and Table 2, respectively, where each predicate $f(x)$ is indicated with a starting capital letter (e.g., Mammal(x)). Following the notation given in 2.3, rules are divided into categories.

Table 1: Domain knowledge on the *Animals* dataset.

	<p>Type a and b rules:</p> $\forall x \text{ Hair}(x) \vee \text{Mammal}(x)$ $\forall x \text{ Milk}(x) \Rightarrow \text{Mammal}(x)$ $\forall x \text{ Feather}(x) \Rightarrow \text{Bird}(x)$ $\forall x \text{ Fly}(x) \wedge \text{LayEggs}(x) \Rightarrow \text{Bird}(x)$ $\forall x \text{ Mammal}(x) \wedge \text{Meat}(x) \Rightarrow \text{Carnivore}(x)$ $\forall x \text{ Mamal}(x) \wedge \text{PointedTeeth}(x) \wedge \text{Claws}(x) \wedge \text{ForwardEyes}(x) \Rightarrow \text{Carnivore}(x)$ $\forall x \text{ Mammal}(x) \wedge \text{Hoofs}(x) \Rightarrow \text{Ungulate}(x)$ $\forall x \text{ Mammal}(x) \wedge \text{Cud}(x) \Rightarrow \text{Ungulate}(x)$ $\forall x \text{ Mammal}(x) \wedge \text{Cud}(x) \Rightarrow \text{Eventoeod}(x)$ $\forall x \text{ Carnivore}(x) \wedge \text{Tawny}(x) \wedge \text{DarkSpots}(x) \Rightarrow \text{Cheetah}(x)$ $\forall x \text{ Carnivore}(x) \wedge \text{Tawny}(x) \wedge \text{BlackStripes}(x) \Rightarrow \text{Tiger}(x)$ $\forall x \text{ Ungulate}(x) \wedge \text{LongLegs}(x) \wedge \text{LongNeck}(x) \wedge \text{Tawny}(x) \wedge \text{DarkSpots}(x) \Rightarrow \text{Giraffe}(x)$ $\forall x \text{ Blackstripes}(x) \wedge \text{Ungulate}(x) \wedge \text{White}(x) \Rightarrow \text{Zebra}(x)$ $\forall x \text{ Bird}(x) \wedge \neg \text{Fly}(x) \wedge \text{LongLegs}(x) \wedge \text{LongNeck}(x) \wedge \text{Black}(x) \Rightarrow \text{Ostrich}(x)$ $\forall x \text{ Bird}(x) \wedge \neg \text{Fly}(x) \wedge \text{Swim}(x) \wedge \text{BlackWhite}(x) \Rightarrow \text{Penguin}(x)$ $\forall x \text{ Bird}(x) \wedge \text{GoodFlier}(x) \Rightarrow \text{Albatross}(x)$
	<p>Type c rule:</p> $\forall x \text{ mutual_excl}(\text{Albatross}(x), \text{Giraffe}(x), \text{Cheetah}(x), \text{Ostrich}(x), \text{Penguin}(x), \text{Tiger}(x), \text{Zebra}(x))$
	<p>Type d rule:</p> $\forall x \text{ Mammal}(x) \vee \text{Hair}(x) \vee \text{Milk}(x) \vee \text{Feathers}(x) \vee \text{Bird}(x) \vee \text{Fly}(x)$ $\vee \text{LayEggs}(x) \vee \text{Meat}(x) \vee \text{Carnivore}(x) \vee \text{PointedTeeth}(x) \vee \text{Claws}(x) \vee \text{ForwardEyes}(x)$ $\vee \text{Hoofs}(x) \vee \text{Ungulate}(x) \vee \text{Cud}(x) \vee \text{Eventoeod}(x) \vee \text{Tawny}(x) \vee \text{BlackStripes}(x)$ $\vee \text{LongLegs}(x) \vee \text{LongNeck}(x) \vee \text{DarkSpots}(x) \vee \text{White}(x) \vee \text{Black}(x) \vee \text{Swim}(x)$ $\vee \text{BlackWhite}(x) \vee \text{GoodFlier}(x)$

Table 2: Domain knowledge on the *PASCAL-Part* dataset.

Type a rules:	
$\forall x$	$\text{Screen}(x) \Rightarrow (\text{Tvmonitor}(x))$
$\forall x$	$\text{Coach}(x) \Rightarrow (\text{Train}(x))$
$\forall x$	$\text{Torso}(x) \Rightarrow (\text{Person}(x) \vee \text{Horse}(x) \vee \text{Cow}(x) \vee \text{Dog}(x) \vee \text{Bird}(x) \vee \text{Cat}(x) \vee \text{Sheep}(x))$
$\forall x$	$\text{Leg}(x) \Rightarrow (\text{Person}(x) \vee \text{Horse}(x) \vee \text{Cow}(x) \vee \text{Dog}(x) \vee \text{Bird}(x) \vee \text{Cat}(x) \vee \text{Sheep}(x))$
$\forall x$	$\text{Head}(x) \Rightarrow (\text{Person}(x) \vee \text{Horse}(x) \vee \text{Cow}(x) \vee \text{Dog}(x) \vee \text{Bird}(x) \vee \text{Cat}(x) \vee \text{Sheep}(x))$
$\forall x$	$\text{Ear}(x) \Rightarrow (\text{Person}(x) \vee \text{Horse}(x) \vee \text{Cow}(x) \vee \text{Dog}(x) \vee \text{Cat}(x) \vee \text{Sheep}(x))$
$\forall x$	$\text{Eye}(x) \Rightarrow (\text{Person}(x) \vee \text{Cow}(x) \vee \text{Dog}(x) \vee \text{Bird}(x) \vee \text{Cat}(x) \vee \text{Horse}(x) \vee \text{Sheep}(x))$
$\forall x$	$\text{Eyebrow}(x) \Rightarrow (\text{Person}(x))$
$\forall x$	$\text{Mouth}(x) \Rightarrow (\text{Person}(x))$
$\forall x$	$\text{Hair}(x) \Rightarrow (\text{Person}(x))$
$\forall x$	$\text{Nose}(x) \Rightarrow (\text{Person}(x) \vee \text{Dog}(x) \vee \text{Cat}(x))$
$\forall x$	$\text{Neck}(x) \Rightarrow (\text{Person}(x) \vee \text{Horse}(x) \vee \text{Cow}(x) \vee \text{Dog}(x) \vee \text{Bird}(x) \vee \text{Cat}(x) \vee \text{Sheep}(x))$
$\forall x$	$\text{Arm}(x) \Rightarrow (\text{Person}(x))$
$\forall x$	$\text{Muzzle}(x) \Rightarrow (\text{Horse}(x) \vee \text{Cow}(x) \vee \text{Dog}(x) \vee \text{Sheep}(x))$
$\forall x$	$\text{Hoof}(x) \Rightarrow (\text{Horse}(x))$
$\forall x$	$\text{Tail}(x) \Rightarrow (\text{Horse}(x) \vee \text{Cow}(x) \vee \text{Dog}(x) \vee \text{Bird}(x) \vee \text{Sheep}(x) \vee \text{Cat}(x) \vee \text{Aeroplane}(x))$
$\forall x$	$\text{Bottle Body}(x) \Rightarrow (\text{Bottle}(x))$
$\forall x$	$\text{Paw}(x) \Rightarrow (\text{Dog}(x) \vee \text{Cat}(x))$
$\forall x$	$\text{Aeroplane Body}(x) \Rightarrow (\text{Aeroplane}(x))$
$\forall x$	$\text{Wing}(x) \Rightarrow (\text{Aeroplane}(x) \vee \text{Bird}(x))$
$\forall x$	$\text{Wheel}(x) \Rightarrow (\text{Aeroplane}(x) \vee \text{Car}(x) \vee \text{Bicycle}(x) \vee \text{Bus}(x) \vee \text{Motorbike}(x))$
$\forall x$	$\text{Stern}(x) \Rightarrow (\text{Aeroplane}(x))$
$\forall x$	$\text{Cap}(x) \Rightarrow (\text{Bottle}(x))$
$\forall x$	$\text{Hand}(x) \Rightarrow (\text{Person}(x))$
$\forall x$	$\text{Frontside}(x) \Rightarrow (\text{Car}(x) \vee \text{Bus}(x) \vee \text{Train}(x))$
$\forall x$	$\text{Rightside}(x) \Rightarrow (\text{Car}(x) \vee \text{Bus}(x) \vee \text{Train}(x))$
$\forall x$	$\text{Roofside}(x) \Rightarrow (\text{Car}(x) \vee \text{Bus}(x) \vee \text{Train}(x))$
$\forall x$	$\text{Backside}(x) \Rightarrow (\text{Car}(x) \vee \text{Bus}(x) \vee \text{Train}(x))$
$\forall x$	$\text{Leftside}(x) \Rightarrow (\text{Car}(x) \vee \text{Train}(x) \vee \text{Bus}(x))$
$\forall x$	$\text{Door}(x) \Rightarrow (\text{Car}(x) \vee \text{Bus}(x))$
$\forall x$	$\text{Mirror}(x) \Rightarrow (\text{Car}(x) \vee \text{Bus}(x))$
$\forall x$	$\text{Headlight}(x) \Rightarrow (\text{Car}(x) \vee \text{Bus}(x) \vee \text{Train}(x) \vee \text{Motorbike}(x) \vee \text{Bicycle}(x))$
$\forall x$	$\text{Motorbike}(x) \Rightarrow (\text{Wheel}(x) \vee \text{Headlight}(x) \vee \text{Handlebar}(x) \vee \text{Saddle}(x))$
$\forall x$	$\text{Window}(x) \Rightarrow (\text{Car}(x) \vee \text{Bus}(x))$
$\forall x$	$\text{Plate}(x) \Rightarrow (\text{Car}(x) \vee \text{Bus}(x))$
$\forall x$	$\text{Engine}(x) \Rightarrow (\text{Aeroplane}(x))$
$\forall x$	$\text{Foot}(x) \Rightarrow (\text{Person}(x) \vee \text{Bird}(x))$
$\forall x$	$\text{Chainwheel}(x) \Rightarrow (\text{Bicycle}(x))$
$\forall x$	$\text{Saddle}(x) \Rightarrow (\text{Bicycle}(x) \vee \text{Motorbike}(x))$
$\forall x$	$\text{Handlebar}(x) \Rightarrow (\text{Bicycle}(x) \vee \text{Motorbike}(x))$
$\forall x$	$\text{Train Head}(x) \Rightarrow (\text{Train}(x))$
$\forall x$	$\text{Beak}(x) \Rightarrow (\text{Bird}(x))$
$\forall x$	$\text{Pot}(x) \Rightarrow (\text{Pottedplant}(x))$
$\forall x$	$\text{Plant}(x) \Rightarrow (\text{Pottedplant}(x))$
$\forall x$	$\text{Horn}(x) \Rightarrow (\text{Cow}(x) \vee \text{Sheep}(x))$
Type b rules:	
$\forall x$	$\text{Tvmonitor}(x) \Rightarrow (\text{Screen}(x))$
$\forall x$	$\text{Train}(x) \Rightarrow (\text{Coach}(x) \vee \text{Leftside}(x) \vee \text{Train Head}(x) \vee \text{Headlight}(x) \vee \text{Frontside}(x) \vee \text{Rightside}(x) \vee \text{Backside}(x) \vee \text{Roofside}(x))$
$\forall x$	$\text{Person}(x) \Rightarrow (\text{Torso}(x) \vee \text{Leg}(x) \vee \text{Head}(x) \vee \text{Ear}(x) \vee \text{Eye}(x) \vee \text{Eyebrow}(x) \vee \text{Mouth}(x) \vee \text{Hair}(x) \vee \text{Nose}(x) \vee \text{Neck}(x) \vee \text{Arm}(x) \vee \text{Hand}(x) \vee \text{Foot}(x))$
$\forall x$	$\text{Horse}(x) \Rightarrow (\text{Head}(x) \vee \text{Ear}(x) \vee \text{Muzzle}(x) \vee \text{Torso}(x) \vee \text{Neck}(x) \vee \text{Leg}(x) \vee \text{Hoof}(x) \vee \text{Tail}(x) \vee \text{Eye}(x))$
$\forall x$	$\text{Cow}(x) \Rightarrow (\text{Head}(x) \vee \text{Ear}(x) \vee \text{Eye}(x) \vee \text{Muzzle}(x) \vee \text{Torso}(x) \vee \text{Neck}(x) \vee \text{Leg}(x) \vee \text{Tail}(x) \vee \text{Horn}(x))$
$\forall x$	$\text{Bottle}(x) \Rightarrow (\text{Bottle Body}(x) \vee \text{Cap}(x))$
$\forall x$	$\text{Dog}(x) \Rightarrow (\text{Head}(x) \vee \text{Ear}(x) \vee \text{Torso}(x) \vee \text{Neck}(x) \vee \text{Leg}(x) \vee \text{Paw}(x) \vee \text{Eye}(x) \vee \text{Muzzle}(x) \vee \text{Nose}(x) \vee \text{Tail}(x))$
$\forall x$	$\text{Aeroplane}(x) \Rightarrow (\text{Aeroplane Body}(x) \vee \text{Wing}(x) \vee \text{Wheel}(x) \vee \text{Stern}(x) \vee \text{Engine}(x) \vee \text{Tail}(x))$
$\forall x$	$\text{Car}(x) \Rightarrow (\text{Frontside}(x) \vee \text{Rightside}(x) \vee \text{Door}(x) \vee \text{Mirror}(x) \vee \text{Headlight}(x) \vee \text{Wheel}(x) \vee \text{Window}(x) \vee \text{Plate}(x) \vee \text{Roofside}(x) \vee \text{Backside}(x) \vee \text{Leftside}(x))$
$\forall x$	$\text{Bus}(x) \Rightarrow (\text{Plate}(x) \vee \text{Frontside}(x) \vee \text{Rightside}(x) \vee \text{Door}(x) \vee \text{Mirror}(x) \vee \text{Headlight}(x) \vee \text{Window}(x) \vee \text{Wheel}(x) \vee \text{Leftside}(x) \vee \text{Backside}(x) \vee \text{Roofside}(x))$
$\forall x$	$\text{Bicycle}(x) \Rightarrow (\text{Wheel}(x) \vee \text{Chainwheel}(x) \vee \text{Saddle}(x) \vee \text{Handlebar}(x) \vee \text{Headlight}(x))$
$\forall x$	$\text{Bird}(x) \Rightarrow (\text{Head}(x) \vee \text{Eye}(x) \vee \text{Beak}(x) \vee \text{Torso}(x) \vee \text{Neck}(x) \vee \text{Leg}(x) \vee \text{Foot}(x) \vee \text{Tail}(x) \vee \text{Wing}(x))$
$\forall x$	$\text{Cat}(x) \Rightarrow (\text{Head}(x) \vee \text{Ear}(x) \vee \text{Eye}(x) \vee \text{Nose}(x) \vee \text{Torso}(x) \vee \text{Neck}(x) \vee \text{Leg}(x) \vee \text{Paw}(x) \vee \text{Tail}(x))$
$\forall x$	$\text{Motorbike}(x) \Rightarrow (\text{Wheel}(x) \vee \text{Headlight}(x) \vee \text{Handlebar}(x) \vee \text{Saddle}(x))$
$\forall x$	$\text{Sheep}(x) \Rightarrow (\text{Head}(x) \vee \text{Ear}(x) \vee \text{Eye}(x) \vee \text{Muzzle}(x) \vee \text{Torso}(x) \vee \text{Neck}(x) \vee \text{Leg}(x) \vee \text{Tail}(x) \vee \text{Horn}(x))$
$\forall x$	$\text{Pottedplant}(x) \Rightarrow (\text{Pot}(x) \vee \text{Plant}(x))$
Type c rules:	
$\forall x$	$\text{Tvmonitor}(x) \vee \text{Train}(x) \vee \text{Person}(x) \vee \text{Boat}(x) \vee \text{Horse}(x) \vee \text{Cow}(x) \vee \text{Bottle}(x) \vee \text{Dog}(x) \vee \text{Aeroplane}(x) \vee \text{Car}(x) \vee \text{Bus}(x) \vee \text{Bicycle}(x) \vee \text{Table}(x) \vee \text{Chair}(x) \vee \text{Bird}(x) \vee \text{Cat}(x) \vee \text{Motorbike}(x) \vee \text{Sheep}(x) \vee \text{Sofa}(x) \vee \text{Pottedplant}(x)$
Type d rules:	
$\forall x$	$\text{Screen}(x) \vee \text{Coach}(x) \vee \text{Torso}(x) \vee \text{Leg}(x) \vee \text{Head}(x) \vee \text{Ear}(x) \vee \text{Eye}(x) \vee \text{Eyebrow}(x) \vee \text{Mouth}(x) \vee \text{Hair}(x) \vee \text{Nose}(x) \vee \text{Neck}(x) \vee \text{Arm}(x) \vee \text{Muzzle}(x) \vee \text{Hoof}(x) \vee \text{Tail}(x) \vee \text{BottleBody}(x) \vee \text{Paw}(x) \vee \text{AeroplaneBody}(x) \vee \text{Wing}(x) \vee \text{Wheel}(x) \vee \text{Stern}(x) \vee \text{Cap}(x) \vee \text{Hand}(x) \vee \text{Frontside}(x) \vee \text{Rightside}(x) \vee \text{Door}(x) \vee \text{Mirror}(x) \vee \text{Headlight}(x) \vee \text{Window}(x) \vee \text{Plate}(x) \vee \text{Roofside}(x) \vee \text{Backside}(x) \vee \text{Leftside}(x) \vee \text{Engine}(x) \vee \text{Foot}(x) \vee \text{Chainwheel}(x) \vee \text{Saddle}(x) \vee \text{Handlebar}(x) \vee \text{TrainHead}(x) \vee \text{Beak}(x) \vee \text{Pot}(x) \vee \text{Plant}(x) \vee \text{Horn}(x)$