How much can experimental cost be reduced in active learning of agent strategies?

Céline Hocquette and Stephen Muggleton

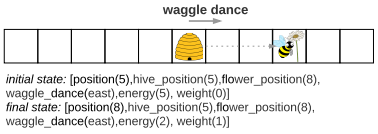
Department of Computing, Imperial College London, UK
{celine.hocquette16,s.muggleton}@imperial.ac.uk

Abstract. In science, experiments are empirical observations allowing for the arbitration of competing hypotheses and knowledge acquisition. For a scientist that aims at learning an agent strategy, performing experiments induces energy costs. To that extent, the efficiency of a learning process relies on the number of experiments performed. We study in this article how the cost of experimentation can be reduced with active learning to learn efficient agent strategies. We consider an extension of the meta-interpretive learning framework that allocates a Bayesian posterior distribution over the hypothesis space. At each iteration, the learner queries the label of the instance with maximum entropy, it is the most discriminative over the remaining competing hypotheses, and thus achieves the highest shrinkage of the version space. We study the theoretical framework and evaluate the gain on the cost of experimentation for the task of learning regular grammars and agent strategies: our results demonstrate that the number of experiments to perform to reach an arbitrary accuracy level can at least be halved.

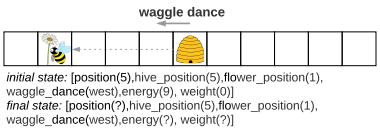
Keywords: Bayesian meta-interpretive learning, active learning, agent-based modelling.

1 Introduction

Once a honeybee has found a rich source of pollen, it shares its location with other members of the colony by executing a particular figure called waggle dance [16]. It guides the search for other bees toward flowers yielding nectar and pollen and thus enhances the efficiency of the colony foraging strategy.
More broadly, strategies are general programs aimed at achieving a goal and that can provide plans for a multiplicity of initial states. When a scientist models animal behaviours or other strategies, the learning process generally requires the realisation of many experiments which set-up is resource exhausting and time consuming. Thus, the learning efficiency relies on the number of performed experiments. We investigate in this work how much the experimental cost can be reduced with active learning to learn agent strategies. An active learner is allowed to actively choose the experiments to perform to acquire knowledge during the learning process. Furthermore, in real-world situations, strategies should be resource-efficient to be beneficial for agents. Therefore, we additionally want



(a) First observation: the bee starts at the hive with no weight carried and ends up at the flower carrying pollen



(c) Second observation, no matter its outcome, it is discriminative for the competing hypotheses of Figure 1b

Hypothesis 1	Hypothesis 2

`f(A,B):-f1(A,C),grab(C,B).`
`f1(A,B):-until(A,B,at_flower,move_right).`
	`f(A,B):-f2(A,C),grab(C,B).`
	`f2(A,B):-until(A,B,at_flower,f1).`
	`f1(A,B):-ifthenelse(A,B,waggle_east,move_right,move_left).`

(b) Two competing hypotheses for the first observation of Figure 1a

Fig. 1: Observations of a bee behaviour

to converge toward efficient strategies.

In section 6, we learn a general strategy for a bee to find pollen in an environment. Learnt strategies are logic programs built from observations of bee behaviour. Observations are labelled as positive if the goal is fulfilled and negative otherwise. Figure 1a represents a positive observation: the waggle dance indicates that a flower is at the right of the hive, the bee flies in this direction and finds pollen. Several hypotheses can be inferred from it, among them the two represented in Figure 1b. To discriminate between them, the experiment of Figure 1c could be performed. The flower is now on the left, which is indicated by the waggle dance. No matter its outcome, positive or negative, it would eliminate one of these two hypotheses. Therefore, it is an informative query.

Meta-Interpretive Learning (MIL) has been demonstrated to be a suitable paradigm to learn strategies since it supports predicate invention and the learning of recursive programs [22,24]. Given the observations so far, consistent hypotheses are built from a set of metarules and the background knowledge. A Bayesian posterior distribution is implemented over the hypothesis space [23] and introduces a bias toward hypotheses with lowest complexity. The learner computes at each iteration the entropies of the possible experiments given the current hypothesis space and the prior distribution. The instance with maximum entropy is selected: it is the most discriminative between the remaining competing hypotheses. This process is resumed and more experiments are performed until some target accuracy is reached.

Specifically, our contributions are the introduction of a framework for learning efficient agent strategies with reduced cost of experimentation and the description of its implementation. We evaluate theoretically the expected gain in entropy. We also demonstrate experimentally that Bayesian MIL Active Learning converges faster toward efficient strategies than a passive learner in the same conditions.

This article is organised as follows. Section 2 describes some related work. Section 3 describes the framework used in this paper together with the learning protocol. Section 4 details a theoretical analysis. Section 5 describes the implementation. Section 6 reports experiments in learning regular grammars and bee strategies. Finally, we conclude and discuss further work in Section 7.

2 Related Work

Active Learning Active Learning is a protocol in which the learner is able to choose the data from which it learns by accessing an oracle. It contrasts with passive learning for which the labeled data is selected at random. The objective in active learning is to learn a model with high accuracy while ideally making fewer queries than the number of random data required by a passive learner to achieve the same accuracy level. It has been widely studied for identifying classifiers [26] and different query strategy frameworks have been introduced.
In the membership query setting, the learner is allowed to ask for the label of any points of the instance space, even artificially generated ones [1,2].However, newly synthesized instances may be uninterpretable by human oracles. An alternative is stream-based selective sampling: the learner can sample from the instance distribution and decide whether to label or discard each sample instance [12], or directly sample from a subpart of the instance space that is the most informative [4]. We focus in this work on pool-based active learning: the learner has access to a large number of initial unlabelled data points, and to an oracle which can provide the label of any of these points on request [20].
Several measures have been suggested for evaluating the shrinkage of the hypothesis space during the learning process and thus measuring the benefits of active learning over passive learning. The main ones are the diameter of the version space [11,28], the measure of the region of disagreement [13,14], the metric entropy [18] and the size of the version space [21,10] which inspired this paper. We will more specifically operate in a Bayesian setting [12,10] that benefits from a prior distribution over the hypothesis space.
Similarly, the system presented in this article is based upon active learning for devising experiments to rule out hypotheses from the version space. However, our approach is different from the work presented above since we use active learning within the construction of logic programs and for learning agent strategies in a Bayesian context.

Decision Trees A search strategy can be representented by a tree whose internal nodes are experiments and whose leaves are hypotheses: minimizing the number of queries means building a tree of minimum average size. In that case, it has been shown that the perfomances of a greedy strategy are not worst than any other strategy for minimizing the number of label queries [10]. Moreover, the expected depth of any binary decision tree is lower bounded by the entropy of the prior distribution [5].

Combining Active Learning with Inductive Logic Programming In [27], Inductive Logic Programming (ILP) has been combined with Active Learning for two non-classification tasks in natural language processing: semantic parsing and information extraction. Also, a closed loop Machine Learning system for Scientific Discovery applications is described in [3,17]: a robot scientist is introduced, it autonomously proposes and performs a sequence of experiments which

minimises the expected cost of experimentation for converging upon an accurate hypothesis generated with ILP. Conversely, our work aims at learning efficient strategies.

Learning Efficient Strategies A general framework for learning optimal resource complexity robot strategies is presented in [6]. It has been extended in [8] into *Metaopt*, an ILP system that learns minimal cost logic programs by adding a general cost function into the meta-interpreter. By contrast, our work focuses on another aspect of the learning efficiency: we investigate how to reduced experimental costs for learning efficient agent strategies.

Relational Reinforcement Learning A challenge in reinforcement learning is the exploration / exploitation trade-off. In [19], the authors present relational exploration strategies: the generalisation of learnt knowledge over unobserved instances in relational worlds allows a generalisation of the notion of known states compared to propositional settings. It can also applied to largest domains. In [25], Active Learning is used to select actions to perform for reaching states that will enforce a revision of the current model. It is shown that the integration of Active Learning improves learning speed: an accurate action model is obtained after performing much less actions than when using random exploration only.

To the authors' best knowledge, this is the first time active learning is integrated with Bayesian MIL to devise a sequence of experiments to perform for learning efficient strategies with reduced experimental costs.

3 Theoretical Framework

3.1 Notations

Let $\mathcal{X}$ be the instance space, and $\mathcal{H}$ a concept class over the instance space $\mathcal{X}$. We consider $\Pi_{\mathcal{X}}$ the probability distribution over the instance space $\mathcal{X}$ and $\Pi_{\mathcal{H}}$ the probability distribution over the hypothesis space $\mathcal{H}$. We assume that the target hypothesis $\bar{H}$ is drawn from $\mathcal{H}$ and according to $\Pi_{\mathcal{H}}$. We call $E_m = \{e_0, ..., e_m\}$ the set of examples selected up to the iteration m. The version space V_m is the set of hypotheses $H \in \mathcal{H}$ consistent with E_m, therefore $V_m \subset \mathcal{H}$.

3.2 Meta-Interpretive Learning (MIL)

MIL is a form of ILP [24,22]. The learner is given a set of examples E and a background knowledge B composed of a set of Prolog definitions B_p and metarules M such that $B = B_p \cup M$. The aim is to generate a hypothesis H such that $B, H \models E$. The proof is based upon an adapted Prolog meta-interpreter. It first attempts to prove the examples considered deductively. Failing this, it unifies the head of a metarule with the goal, and saves the resulting meta-substitution. The body and then the other examples are similarly proved. The meta-substitutions

recovered for each successful proofs are saved and can be used into further proofs by substituting them into their corresponding metarules.
For example, the first clause of the learned hypotheses of Figure 1 `f(A,B):-f1(A,C),grab(C,B).` has been derived from the metarule *chain rule* detailed in Figure 2a by applying the meta-substitution `[f/2,f1/2,grab/2]`.
Two key features of MIL are that it supports predicate invention and the learning of recursive programs. The former enables decomposition of the learned logic program into new sub-actions. The latter allows to learn more general programs and with shorter lengths. Both makes MIL well suited for learning strategies.
The choice of metarules induces a declarative bias on the hypothesis space since it determines the structure of learnable programs: an appropriate choice helps minimising the number of clauses in the consistent hypothesis [7]. Also, the use of higher-order Abstractions supports learning more compact programs [9]. We focus in this work on learning logic programs built from the metarules *chain rule*, *precondition* and *postcondition*, whose description is available in the Figure 2a. Indeed, this set of metarules is enough to learn the class of dyadic logic programs investigated in this paper. The first experiment tackles the task of learning regular grammars. As shown on the Figure 2b, metarules for finite state acceptors can be expressed with *chain rule* and *postcondition* only. The second experiment considers the task of learning agent strategies. Fluents are treated as monadic predicates which apply to a situation, while actions are dyadic predicates which transform one situation to another. We will use $Metagol_{AI}$ which supports Abstractions and Inventions [9]. We use the two abstractions *until/4* and *ifthenelse/5* to reduce the complexity of the learned programs: *until/4* represents a recursive call to the action Ac while some condition $Cond$ is not fullfilled and *ifthenelse/5* expresses a choice between the actions $Then$ and $Else$ based upon the realisation of the condition $Cond$ (Figure 2b). Similarly, these Abstractions can be expressed with the *chain rule*, *precondition* and *postcondition* only as shown in the Figure 2b.

3.3 Complexity of an Hypothesis

The hypotheses generated with MIL differ by their complexity. We distinguish two notions to evaluate the complexity of a logic program H. The textual complexity relies on Occam's principle and represents the length $l(H)$ of H measured as the number of clauses. However, textually smaller programs are not necessarily the more efficient. The resource complexity $r(H)$ of an agent strategy [6] represents the amount of resources (eg: energy) consumed by the agent while executing the strategy. In the following, we will combine the textual complexity with the resource complexity to learn efficient strategies in terms of a global complexity:

$$c(H) = l(H) + r(H)$$

Name	Metarule
chain rule	$P(A,B) \leftarrow Q(A,C), R(C,B).$
precondition	$P(A,B) \leftarrow Q(A), R(A,B).$
postcondition	$P(A,B) \leftarrow Q(A,B), R(B).$

(a) General metarules

Exp.	Name	Metarule
1	*acceptor*	$Q(A,B) \leftarrow eq(A,B), acceptor(B).$
	delta	$Q0(A,B) \leftarrow zero(A,C), Q1(C,B).$
		$Q0(A,B) \leftarrow one(A,C), Q1(C,B).$
2	*until*	$until(A,B,Cond,Ac) \leftarrow Ac(A,B), Cond(B).$
		$until(A,B,Cond,Ac) \leftarrow F(A,C), until(C,B,Cond,Ac).$
	ifthenelse	$Ifthenelse(A,B,Cond,Then,Else) \leftarrow Cond(A), Then(A,B).$
		$Ifthenelse(A,B,Cond,Then,Else) \leftarrow Else(A,C), eq(C,B).$

(b) Metarules used in the experiments

Fig. 2: Metarules considered: the class of dyadic logic program studied in this article can be expressed with chain rules, preconditions and postconditions only

3.4 Bayesian Prior Distribution

The preference for hypotheses with lowest complexity is encoded in a prior distribution which induces a bias over the hypotheses space and favors more efficient strategies. We consider the framework described in [23]. A Bayesian prior probability is defined for any H in $\mathcal{H}$ from the complexity $c(H)$ as follows and for $\frac{1}{a} = \sum_{i=1}^{\infty} \frac{1}{i^2} = \frac{\pi^2}{6}$ being a normalisation constant:

$$\mathit{\Pi}_{\mathcal{H}}(\{H \mid c(H) = k\}) = \frac{a}{k^2}$$

Moreover, given a background knowledge B and a set of examples E, the likelihood of E is:

$$p(E \mid B, H) = \begin{cases} 1 \text{ if } B, H \models E \\ 0 \text{ else} \end{cases}$$

According to Bayes's theorem, the posterior is given by:

$$p(H \mid B, E) = \frac{\mathit{\Pi}_{\mathcal{H}}(H) p(E \mid B, H)}{c}$$

The denominator c is a normalization constant. Therefore, the posterior $p(H \mid B, E)$ is proportional to the prior $\mathit{\Pi}_{\mathcal{H}}(H)$. The MAP hypothesis H_{MAP} is defined as $H_{MAP} = \underset{H}{\text{argmax}}(p(H \mid B, E))$.

3.5 Active Learning

A set of N instances is initially sampled from $\mathcal{X}$. The active learner conducts at each iteration $m+1$ an experiment in which it chooses the next instance e_{m+1} among this set and observes its label returned by an oracle. This information

helps discriminating between the competing hypotheses built as described previously since it rules out some proportion of the version space V_m that is not consistent with it. The shrinkage of the hypothesis space is measured by the ratio $\frac{\Pi_{\mathcal{H}}(V_{m+1})}{\Pi_{\mathcal{H}}(V_m)}$. We associate each sampled instance e with a probability p_e given the version space:

$$p(e) = \frac{\min(\Pi_{\mathcal{H}}(\{H \in V_m \mid H(e) = 1\}), \Pi_{\mathcal{H}}(\{H \in V_m \mid H(e) = 0\})}{\Pi_{\mathcal{H}}(V_m)}$$

This value represents the minimal reduction ratio over the version space induced by the query of the instance e. Moreover, it was noted in [21] that in general, the optimal query strategy is to select an instance covered by half of the the version space. Indeed, no matter its true label, it would halve the size of the version space. Therefore, the query strategy chosen is to select the instance e_m for which $p(e)$ is the closest to $\frac{1}{2}$, that is for which the entropy $ent(p(e))$ is maximal:

$$e_m = \underset{e}{\operatorname{argmax}}(ent(p_e))$$

$$ent(p(e)) = -p(e)log(p(e)) - (1 - p(e))log(1 - p(e))$$

In that case, it is the most informative instance from the learner's point of view, since it is the most discriminative given the current version space.
From an information-theory point of view, the expected entropy of $p(e)$ is the expected information gain from the label of e_m [15].

3.6 Learning Protocol

The learning protocol is summarised in the Figures 3 and 4 and represents how the learner acquires information. First, a pool of N instances is randomly sampled. The training set is initialised with one positive instance randomly selected. At each iteration, a fresh new set of K hypotheses consistent with the examples of the training set is sampled. The entropy of each training instance is computed from the set of sampled hypotheses and the instance with maximum entropy is selected. The oracle provides its label, and it is added to the training set. This process is resumed until the maximum number of iterations is reached.

```
Inputs: oracle O, N, K, I
1. Sample N instances from the instance space
2. Initialization: randomly select a positive initial instance
3. While the number of experiments is lower than I:
 - Sample K hypotheses from the hypotheses space
 - Select the instance with maximum entropy
 - Query its label to the oracle O and add it to the training set
Output: hypothesis H with the lowest complexity from the sampled set
```

Fig. 3: Pseudo-code of the framework studied

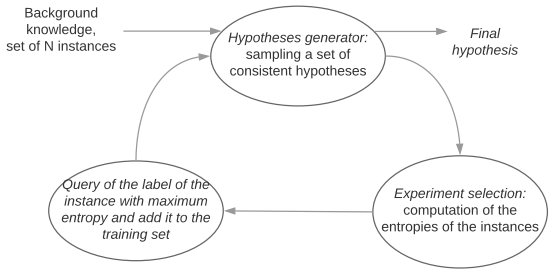


Fig. 4: Diagram of the framework studied: active learning is integrated within the MIL framework

4 Theoretical Analysis

We evaluate the instantaneous expected gain on entropy, which represents the expected reduction of the hypothesis space at one iteration. We assume that the target hypothesis $\bar{H}$ is drawn from $\mathcal{H}$. Each instance e from $\mathcal{X}$ is associated a probability $p(e)$. We consider an arbitrary probability distribution $\mathcal{D}$ over $\mathcal{X}$ at some iteration m, that is bounded in $[0, \frac{1}{2}]$.

Lemma 1. *A set of N instances $\{x_1, ...x_N\}$ is randomly sampled from the instance space $\mathcal{X}$. The active learner selects the instance x_i with maximum entropy among this sample set of size N. Then, the probability of selecting an instance with maximal entropy on $\mathcal{D}$ is N times the one of a passive learner in the same conditions.*

Proof. Let's take $\epsilon > 0$, p_ϵ is set to the probability number in $[0, \frac{1}{2}]$ such that a ϵ-proportion of the instance space has a probability greater or equal to p_ϵ. Let's call $p(x_i)$ the probability of the instance selected. Then $p(x_i) < p_\epsilon$ if and only if every instance from the sample set has a probability smaller than p_ϵ. The instances being independently sampled, it can be written as following:

$$p(p(x_i) < p_\epsilon) = p(p(x_1) < p_\epsilon, ..., p(x_N) < p_\epsilon) = \prod_{k=1}^{N} p(p(x_k) < p_\epsilon) = (1-\epsilon)^N$$

Then, the probability for an active learner to select an instance with probability at least p_ϵ is, from the binomial theorem:

$$p_{active}(p(x_i) \geq p_\epsilon) = 1 - (1-\epsilon)^N = 1 - \sum_{k=0}^{N} \binom{n}{k} (-\epsilon)^k = N\epsilon - o(\epsilon)$$

By comparison, the probability for a passive learner to select an instance with probability at least p_ϵ simply is:

$$p_{passive}(p(x_i) \geq p_\epsilon) = \epsilon$$

Therefore, the probability of selecting an instance with maximal entropy on $\mathcal{D}$ is N times bigger for the active learner.

5 Implementation

5.1 Sampling a Set of Hypotheses

To cope with very large or potentially infinite hypothesis spaces, a set of consistent hypotheses is sampled at each iteration. This sample set is used both to measure the accuracy and to evaluate the entropies.
We use a process called Regular Sampling [23] which limits the number of duplicates while maintaining a good sampling efficiency. A set of probability fractions p_i is generated from the first K integers and with the following two properties: it is evenly distributed in $[0, 1]$ and it is isomorphic to $\mathbb{N}$ for K infinite. Consistent hypotheses are ordered in SLD order within the derivation tree. Samples are selected from the tree leafs and following the probability fractions generated. At each node of the tree, the different branches are given equal weights, and are associated a cumulative posterior probability computed as the sum of the posterior probabilities of the hypotheses on the left side. Starting from the top node, we browse through the tree and selects at each node the branch whose cumulative posterior probability interval $[min, max]$ contains the sampling probability fraction p_i. This latter is then updated as $(p_i - min)(max - min)$, and the process is repeated within the sub-tree selected. At the end, Regular Sampling reproduces sampling without replacement due to the distribution of the sequence of fractions, and provides a sample set representative of the current version space. A set of at most K hypotheses is dynamically sampled according to this process. The first K fractions and corresponding hypotheses are generated. If all of them are inconsistent with the examples selected so far, a new set of hypotheses is sampled from the next K natural numbers, and so forth until at least one consistent hypothesis is returned. After removing potential duplicates, this sampled set is saved for evaluating the accuracy and the entropies.

5.2 Computing the Entropies

As a next step, the entropies are computed from the sampled hypotheses. For every instance initially sampled from the instance space, we compute the proportion of sampled hypotheses that predict a positive label and weight it according to the hypotheses prior distribution. The entropy is derived from this probability. Finally, the instance with the highest entropy is selected. If several instances reach the maximum, one of them is selected at random.

6 Experiments

6.1 Experimental Hypothesis

This section describes two experiments for evaluating the benefits of Bayesian Meta-Interpretive Active Learning over the speed of convergence when learning efficient agent strategies[1]. Thus, we investigate the following research hypothesis:

[1] Code for these experiments available at https://github.com/celinehocquette/Bayesian-MIL-active-learning.git

```
q0([0|A],B) :- q1(A,B).
q0([1|A],B) :- q0(A,B).
q1([0|A],B) :- q0(A,B).
q1([1|A],B) :- q1(A,B).
q0([],[]).
```

Target hypothesis

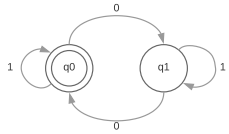


Fig. 5: Example of target hypothesis learned and corresponding FSA: the parity grammar

Research Hypothesis: *Bayesian Meta-Interpretive Active Learning requires a smaller sample complexity for learning efficient agent strategies.*

For the sake of comparison, we consider a passive learner which randomly selects one instance at each iteration. Therefore, we associate to the previous research hypothesis the following null hypothesis that we will test:

Null Hypothesis: *Bayesian MIL Active Learning can not converge faster toward efficient strategies than Bayesian MIL Passive learning.*

6.2 Learning Regular Grammars

We learn regular languages, which are equivalent to deterministic Finite State Automata (FSA). Generally speaking, FSA represent sequences of actions depending on a sequence of events and an input state. Thus, they consist of compact ways of representing strategies. We additionally require target grammars to have a generality $g(H) = \Pi_{\mathcal{X}}(\{x \mid H(x) = 1\})$ between $\frac{1}{3}$ and $\frac{2}{3}$ such that the initial probability for an instance to be positive is $\frac{1}{2}$ on average. This also ensures that trivial grammars are not considered. An example of a target hypothesis and its corresponding automaton are represented on Figure 5: it accepts any string with an even number of 0.

Material and Methods Target grammars are generated with *Metagol*, from a set of sequences regularly sampled from Σ^* and for $\Sigma = \{0, 1\}$. The number of states $n \geq 3$ is generated according to an exponential decay distribution with mean 4. The generality of the hypothesis returned is measured against a set of 40 new regularly sampled instances. These steps are repeated until a grammar with generality in $[\frac{1}{3}; \frac{2}{3}]$ is found. A new number of states is similarly generated to bound the search space.
The metarules provided are *acceptor/1* and *delta/3* described previously in Figure 2. The complexity of the hypotheses is set to their length $l(H)$, and the prior is computed by $\frac{1}{l(H)^2}$. For each target grammar, 150 training instances are initially regularly sampled from Σ^*: a threshold on the probability fraction used for sampling is randomly generated for each instance, thus their length is bounded. Similarly, another 50 instances are sampled for testing.

```
f(A,B):- f2(A,C),grab(C,B).
f2(A,B):- until(A,B,at_flower,f1).
f1(A,B):- ifthenelse(A,B,waggle_east,move_right,move_left).
```

Target hypothesis

Fig. 6: Target hypothesis for the bee experiment

At each iteration, 50 hypotheses are regularly sampled. The accuracy is measured as the average accuracy of all sampled hypotheses over the testing set. The results are presented in the Figure 7 and have been averaged over 50 trials.

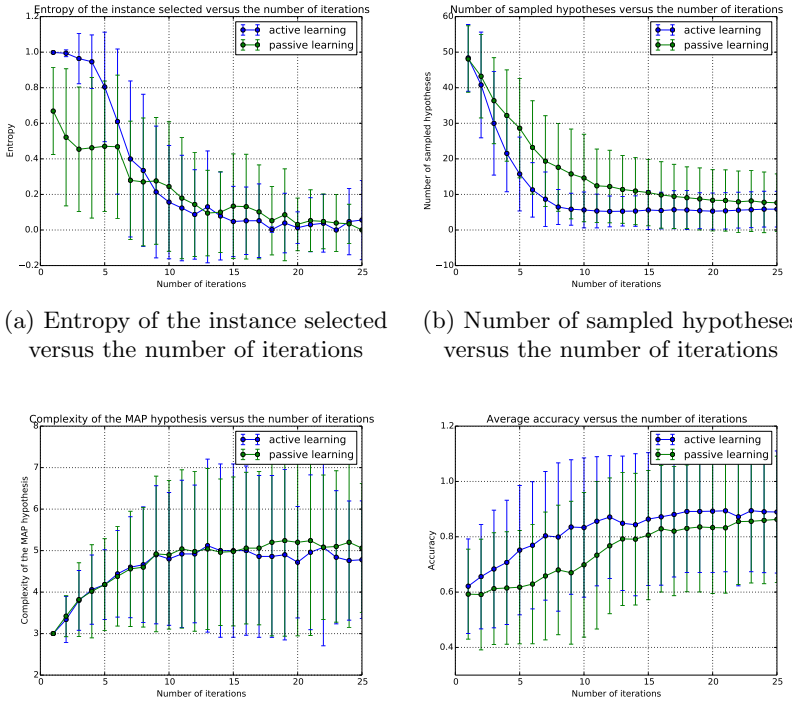
6.3 Learning a Bee Strategy

We learn the strategy introduced in section 1 and that describes a bee strategy for finding pollen following information given by a waggle dance. The target strategy is represented in the Figure 6: until the bee reaches the flower, it flies in the direction given by the waggle dance, and then grabs pollen.

Material and Methods The world is a one-dimensional space of size 10. The state of the world is described by a list of facts. Actions are dyadic predicates that modify the state of the world. The primitive actions are the following: *move_right/2*, *move_left/2*, *grab/2*, they all have a cost of one unit of energy. The metarules used are the chain rule and two higher-order abstractions *until/4* and *ifthenelse/5*, they have been described previously on Figure 2. For any hypothesis H with length $l(H)$, the resource complexity $r(H)$ is measured against the examples selected so far, and the prior of H is then defined as $\frac{1}{l(H)+r(H)}$. The maximum length of an hypothesis is set to 3. The hive is located in the middle position of the environment. A flower is randomly positioned, and a waggle dance indicates if it is east or west of the hive. In the initial state, the bee is at the hive with no pollen carried. It has some amount of energy randomly generated between 0 and 30. In the final state, it is on the flower with one or zero unit of pollen carried. Positive examples are pairs of states for which the task of finding pollen is fulfilled, and with a positive amount of energy in the final state. Negative examples are any pairs of states for which the task is not fulfilled or resulting in a negative amount of energy in the final state. Training and test sets are respectively made of 20 and 40 examples, half positive and half negative. The results are presented in the Figure 8 and have been averaged over 20 trials.

6.4 Results and Discussion

The results are presented on Figures 7 and 8. The learning process takes between 10 minutes and a couple of hours for the grammar experiment according to the complexity of the target hypothesis, and around a few seconds for each run for the bee experiment. The entropy (Figure 7a and 8a) is smaller and less regular for passive learning, which is above all visible for a small number of iterations



(a) Entropy of the instance selected versus the number of iterations

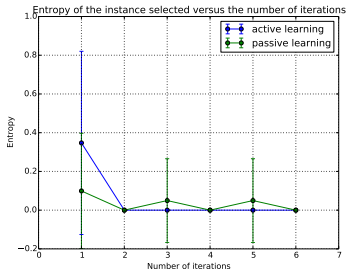
(b) Number of sampled hypotheses versus the number of iterations

(c) Complexity of the sampled hypothesis with highest prior

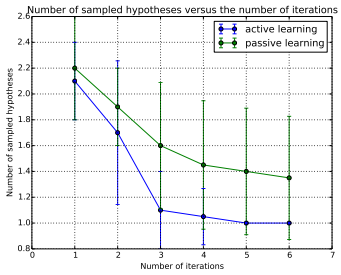
(d) Accuracy versus the number of iterations

Fig. 7: Learning a regular grammar with Bayesian MIL: comparison between active and passive learning; the convergence is faster for active learning.

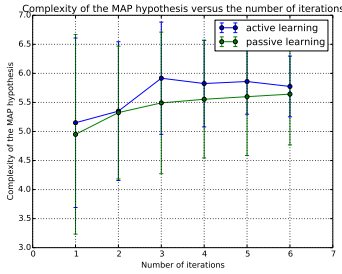
(smaller than 10). In both cases the entropy is globally decreasing as the version space shrinks. The difference between the two curves represents the gain over the reduction of the version space. However, the entropy is smaller to 1 and thus the number of hypotheses does not decrease by a factor of two as in the ideal case, even for active learning. The number of sampled hypotheses is represented on Figures 7b and 8b, it is decreasing with the number of iterations, eventually converging to one hypothesis. The decay rate gets smaller as the entropy drops. The complexity of the MAP hypothesis (Figure 7c and 8c) increases with the number of iterations both for passive and active learning. Indeed, the search is conducted such that the smallest hypotheses that are consistent with the examples are preferred. Therefore, the prior of the MAP hypotheses is increasing. Finally, the accuracy (Figure 7d and 8d) increases, starting between 0.6 and 0.7 (the default accuracy is around 0.5, and the learning process starts with one positive instance for initialisation) and converges toward 1. The convergence is longer and not guaranteed for the grammar experiment, since the hypothesis space is bounded: a number of states maximum is generated before the learning. Figure 9 compares the number of queries required to reach some accuracy level



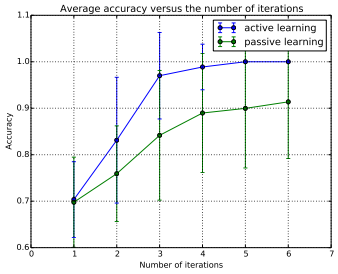
(a) Entropy of the instance selected versus the number of iterations



(b) Number of sampled hypotheses versus the number of iterations



(c) Complexity of the sampled hypothesis with highest prior



(d) Accuracy versus the number of iterations

Fig. 8: Learning a bee strategy with Bayesian MIL: comparison between active and passive learning; the convergence is faster for active learning.

for active and passive learning, it suggests that less iterations are required to achieve good performances, and that experimental costs can at least be halved with active learning. A Mann-Whitney U test indicates that the results are significant at a 0.01 level. According to Lemma 1, the size N of the pool of instance initially sampled should be big enough to ensure that instances with high entropy can always be found. Increasing this number should lift up the entropy of the instance selected, and therefore accelerate the convergence. The number K of hypotheses sampled should be big enough to ensure that the sample set is representative of the hypothesis space, and thus relies on the size of the version space.

Accuracy	Active learning	Passive learning
0.75	5	12
0.80	7	15
0.85	11	22

(a) FSA

Accuracy	Active learning	Passive learning
0.80	2	3
0.90	3	6

(b) Bee experiment

Fig. 9: Number of iterations required to reach some accuracy level for active and passive learning: experimental costs can be halved with active learning.

Even though these results seem encouraging, the empirical evaluation has been performed on a rather small domain and on artificial problems. We plan to demonstrate the scalability of this approach on a real-world dataset as future work.

7 Conclusion and Future Work

This article extends previous work on Meta-Interpretive Learning by integrating active learning for learning efficient agent strategies. We study how automated experimentation can help reducing the experimental costs required to reach some target accuracy. Mainly, we show over two examples that one can expect to halve the experimental costs with active learning and compared to passive learning. We believe that this approach is of interest in AI domains such as robotics or agent-based modelling and for a wide range of applications.
A limitation of this article is the lack of theoretical bound on the sample complexity, which we characterise as future work. In the future, we also want to demonstrate the scalability of this approach by considering real-world datasets. A next step is to use this framework to uncover novel logic programs instead of known target hypotheses, to show that it supports scientific knowledge discovery. Another work direction is to improve the sampling process over the instance space. So far, instances are sampled before the learning. We think about generating at each iteration a new sample set of instances, sampled given the knowledge acquired so far. Instances could also eventually be synthesised. Finally, we want to study other query strategies. Experiments are so far the observation of a binary output for a particular set-up, which could be extended to probabilistic observations.

References

1. D. Angluin. Queries and concept learning. *Journal of automated reasoning*, 2(4):319–42, 1988.
2. D. Angluin. Queries revisited. *Theoretical Computer Science*, 313(2):175194, 2004.
3. C.H. Bryant, S.H. Muggleton, S.G. Oliver, D.B. Kell, P. Reiser, and R.D. King. Combining inductive logic programming, active learning and robotics to discover the function of genes. *Electronic Transactions in Artificial Intelligence, 5-B1(012)*, pages 1–36, 2001.
4. D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, May 1994.
5. M. T. Cover and J. A. Thomas. Elements of information theory. Wiley, 2006.
6. A. Cropper and S. H. Muggleton. Learning efficient logical robot strategies involving composable objects. In *IJCAI 2015*, pages 3423–3429.
7. A. Cropper and S. H. Muggleton. Logical minimisation of meta-rules within meta-interpretive learning. In *ILP 2014*, pages 62–75.
8. A. Cropper and S. H. Muggleton. Learning efficient logic programs. *Machine Learning*, 2018.

9. A. Cropper and S.H. Muggleton. Learning higher-order logic programs through abstraction and invention. In *IJCAI 2016*, pages 1418–1424.
10. S. Dasgupta. Analysis of a greedy active learning strategy. *Advances in neural information processing systems*, 17:337344, 2005.
11. S. Dasgupta. Coarse sample complexity bounds for active learning. *Proceedings of the 18th International Conference on Neural Information Processing Systems*, pages 235,242, 2005.
12. Y. Freund, H.S. Seung, E. Shamir, and N. Tishby. Selective samping using the query by committee algorithm. *Machine Learning*, pages 1551–1557, 1997.
13. S. Hanneke. A bound on the label complexity of agnostic active learning. *In ICML*, 2007.
14. S. Hanneke. Theory of disagreement-based active learning. *Foundations and Trends in Machine Learning*, 7, 2014.
15. D. Haussler, M. Kearns, and R. E. Schapire. Bounds on the sample complexity of bayesian learning using information theory and the vc dimension. *Machine Learning, 14*, pages 83–113, 1994.
16. F. Karl von. The dance language and orientation of bees. *The Belknap Press of Harvard University Press., Cambridge, Massachussets*, 1967.
17. R.D. King, K.E. Whelan, F.M. Jones, P.K.G. Reiser, C.H. Bryant, S.H. Muggleton, D.B. Kell, and S.G. Oliver. Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature 427*, pages 247–252, 2004.
18. S.R. Kulkarni, S.K. Mitter, and J.N. Tsitsiklis. Active learning using arbitrary binary valued queries. *Machine Learning*, 11:2335, 1993.
19. T. Lang, M. Toussaint, and K. Kersting. Exploration in relational worlds. In *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part II*, pages 178–194, 2010.
20. D. Lewis and W. Gale. A sequential algorithm for training text classifiers. In *In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12. ACM/Springer, 1994.
21. T. M. Mitchell. Version spaces: An approach to concept learning. *PhD Thesis*, 1978.
22. S.H. Muggleton and D. Lin. Meta-interpretive learning of higher-order dyadic datalog: Predicate invention revisited. *In Proceedings of the 23rd International Joint Conference Artificial Intelligence*, pages 1551–1557, 2013.
23. S.H. Muggleton, D. Lin, J. Chen, and A. Tamaddoni-Nezhad. Metabayes: Bayesian meta-interpretative learning using higher-order stochastic refinement. *Gerson Zaverucha, Vitor Santos Costa, and Aline Marins Paes, editors, Proceedings of the 23rd International Conference on Inductive Logic Programming (ILP 2013), Berlin, Springer-Verlag. LNAI 8812*, pages 1–17, 2014.
24. S.H. Muggleton, D. Lin, N. Pahlavi, and A. Tamaddoni-Nezhad. Meta-interpretive learning: application to grammatical inference. *Machine Learning 94*, pages 25–49, 2014.
25. C. Rodrigues, P. Gérard, C. Rouveirol, and H. Soldano. Active learning of relational action models. In *Proceedings of the 21st International Conference on Inductive Logic Programming*, ILP'11, pages 302–316, Berlin, Heidelberg, 2012. Springer-Verlag.
26. B. Settles. Active learning literature survey. 52, 07 2010.
27. C. A. Thompson, M. E. Califf, and R. J. Mooney. Active learning for natural language parsing and information extraction. In *Proceedings of the 16th International*

Conference on Machine Learning, ICML '99, pages 406–414. Morgan Kaufmann Publishers Inc., 1999.
28. C. Tosh and S. Dasgupta. Diameter-based active learning. *CoRR*, abs/1702.08553, 2017.